Pruebas de Concepto para el Internet de las Cosas

Tutor: Jesús García Herrero
Co-Tutor: Álvaro Luis Bustamante
Autor: Juan Del Castillo Gómez

Por y Para mi Familia;

Índice

1- Introducción

1.1- Introducción

La filosofía de la ciencia, la divina ciencia, estudia la Inteligencia Artificial Bio-Inspirada como aquella disciplina que estudio la simulación, el análisis y la síntesis de comportamientos, sistemas, circuitos y en definitiva estructuras formales. Estructuras con inspiración biológica que sirven principalmente para estudiar lo relativo al mundo humano y el reino animal. Por ejemplo, tenemos estructuras basadas en autómatas celulares y algoritmos de búsqueda de soluciones en diversos espacios de funciones como los genéticos. Otra ejemplo es el aprendizaje ya sea inductivo, por refuerzo, guiado con sistemas de clasificación o híbridos como los de refuerzo e inducción. Además, tenemos los sistemas colectivos de inteligencia en enjambre o en red o colaborativos, basados en las leyes naturales de cooperación y competición al más puro estilo de las teorías evolutivas de los biólogos o zoólogos, por ejemplo con orangutanes, abejas, hormigas, pájaros, manadas, sociedades de lobos, insectos, y todo tipo de célula ejemplificado por el Juego de la Vida de Conway.

La cultura tecnológica actual es masiva y cada vez crecen nuevas generaciones de colectivos que consumen información de las tecnologías de la comunicación y la información, de algún modo son consumidos con el conocimiento que perciben y procesan pero con una medida adecuada y diversos filtros y límites se puede alcanzar un estado de sabiduría de excelencia en una sociedad globalizada y basada en la meritocracia y guiada por el neuro-marketing.

Estos filtros y límites son dependientes del sujeto y de se educación y evolución. Popularmente clasificamos en Makers, Geeks y Nerds a quienes se construyen su propias herramientas y obras, quienes saben mucho sobre tecnología y los que saben mucho de ciencia y tecnología. Además también podemos hablar de los nativos digitales pero no vamos a entrar en el juicio de cada clasificación social estandarizada.

“Do It Yourself” (DIY) es una filosofía de vida que lleva mucha carga de maker pero también de nerd y geek incluso con el tiempo de nativos digitales. Todo son experimentos caseros de bajo coste que se implementan gracias a la dedicación y la curiosidad, a veces en familia.

El IoT (Intenet of Things) añade servicios de inteligencia artificial a la puerta de las casas, en los bolsillos de las personas, en los dedos, la muñeca o el corazón. Es un cambio brutal con el pasado mas reciente, la brecha digital. Este trabajo añade instrumentos virtuales de música al internet de las cosas, con lo que las agrupaciones modernas y profesionales podrán ensayar juntos vía vídeo y audio conferencia cada uno con su instrumento electroacústico virtual.

Otro punto clave es el router que todos tenemos en casa, es el que sabe todo lo que ocurre dentro y fuera y que también añade servicios inteligentes de tratamiento de la información.

Para implementar o desarrollar estos sistemas se utilizan deiferentes lenguajes informáticos así como paradigmas de orientación a objetos, aspectos, datos, fuentes, etc...

Una vez la inteligencia de fuentes abiertas OSINT (OpenSource Intelligence) ha llegado al

mercado del consumidor tenemos tecnología militar desde 1970 hasta ahora disponible en Internet para quién la sepa encontrar. Es una tarea de análisis intensivo que requiere una fuerte especialización.

La robótica humanoide representa esta inteligencia al servicio del ser humano y todo lo que le rodea, ¿Cuidarán de nuestra salud los robots o se enfadarán con el vecino enemigo?

En el presente siglo la sociedad es global, está totalmente informatizada, es la información la que educa a las masas siendo la tecnología transcultural pues transmite el desarrollo en extremo de la técnica desarrollada desde las primeras herramientas primitivas.

Tanto la nueva educación digital como la sociabilidad de todos los sujetos son pilares fundamentales para el desarrollo de las tecnologías de la información y las comunicaciones, TIC.
La educación persigue mentes resolutivas, orientadas a resultados, analistas, pragmáticas, trabajo en grupos especializados o interdisciplinares y finalmente potenciar e incentivar la creatividad y la comunicación. Las redes sociales representan la sociedad de un modo virtual susceptible de ser analizado profundamente para extraer comportamientos de los grupos de individuos, por ejemplo con teoría de redes y de juegos.

Las TIC están fuertemente basadas en la ciencia, sobretodo en las matemáticas, la estadística, las ciencias físicas y la cibernética construyendo avanzados sistemas de información y conocimiento distribuidos a través de todo el planeta e incluso en el universo más cercano.

Como resultado de estos profundos cambios que están ocurriendo en la sociedad post-industrial, el futuro inmediato de las nuevas tecnologías apunta hacia todo tipo de sistemas robóticos, desde los industriales a robots de servicio, asistencia o humanoides.

Tecnológicamente, estamos viviendo una apasionante etapa de convergencia que permite aunar los esfuerzos de toda la globalidad implicada en desarrollar cada tecnología. La tendencia es que todo pase por la estandarización para que la accesibilidad y la productividad vayan unidas hacia un mismo fin.

Por un lado tenemos diseños basados en la independencia de la implementación física o lógica, o sea, hardware o software, que permiten hacer que las ideas puedan convertirse en innovación y que ésta perdure y no sea mera moda. De esta manera lograremos crear un progreso estable con fines conjuntos que se integre en la sociedad desde principios de siglo.

Bajo mi punto de vista, la tecnología es el medio para conseguir un entorno mejor en el que vivir y una vez se haya implantado el elenco de tecnologías conseguiremos un entorno más productivo y competitivo que pueda pasar del eslabón evolutivo de la competición al de la cooperación. Con esto se podrán conseguir cambios basados en la fusión de culturas que conviertan este siglo primero en el de la información, seguido por el del conocimiento y dirigido hacia el del estudio de las ciencias de la vida, las biociencias.

Por otro lado, seguimos teniendo factores pujantes que persiguen implantar su solución en vez de la estándar, ya que el proceso de estandarización es complejo a la vez que continuo y no se entrega al mercado instantáneamente. Así pues, hemos de saber que éstos esfuerzos son los que

promueven perseguir la solución óptima y que a base de ellos se consigue llegar a estándares basados en la meritocracia aunque provengan de entornos mercantiles seguidos por los técnicos y por lo tanto la mercadotecnia sea un factor decisivo.

En este trabajo no se pretende hacer ninguna otra innovación que lo expresado anteriormente. Por lo tanto, se pretende conseguir soluciones que se basen en los estándares o tendencias actuales pero que también aporte una visión particular adaptada al mercado y que pueda ser entregada a éste.

Así hemos decidido llevar a cabo varias ideas enmarcadas dentro de una línea de desarrollo más amplia:

En sentido deductivo, hemos planteado un marco basado en el concepto de Internet de las Cosas que puede ser aplicable a soluciones industriales, domésticas, musicales, sociales...etc, y que se puede expresar como marco de aplicaciones verticales a la realidad física.
Concretando aún más, hemos propuesto varios planteamientos que ilustran la capacidad de la arquitectura en alguna necesidad concreta. Una necesidad, es la de mejorar las interfaces hombre-máquina mediante un sistema o aplicación de comando y control con el habla o voz. Otra necesidad es la de integrar el sistema en servicios robóticos o de inteligencia artificial. Y no hay que olvidar que al ser estos servicios muy sensibles también hemos tenido en cuenta las dimensiones que abarcan la seguridad informática, el software empresarial y la computación intensiva.

Un ejemplo entendible de la aplicación sería conseguir un sistema que mediante comandos de voz a un dispositivo con micrófono y navegador web sea capaz de ejecutar una orden remota para actuar sobre otro dispositivo o software de manera fiable. Por ejemplo, abrir una puerta, encender una luz, ejecutar una aplicación, etc...Hemos de tener claro que los objetivos únicamente son plantear posibles soluciones pero sin llegar a implementarlas al completo.

1.2- Objetivos y Justificación

Así pues, los objetivos del proyecto únicamente serán pequeñas pruebas de concepto que permitirán en el futuro expandirse y generar aplicaciones concretas. Pero, idealmente se debería conseguir una sola aplicación que integre todas las ideas presentadas y que pueda ser usada para diferentes objetivos, por ejemplo domóticos, musicales, industriales o ambientales.

En cualquier caso, vamos a establecer tres objetivos básicos para tener una guía:

1º- Diseñar y desarrollar pruebas de concepto con hardware abierto que dejen abiertas las posibilidades para el futuro desarrollo de aplicaciones o servicios más complejos basados en software embebido en dicho hardware integrado al chasis.

2º- Integrar el sistema embebido en Internet de las Cosas, concretamente en una plataforma ya desarrollada en fase de pruebas que nos brinda la posibilidad de controlar desde un panel de control todas nuestras “cosas” de una forma simplificada.

3º- Dejar planteado un entorno de computación distribuida capaz de soportar aplicaciones horizontales o verticales con algoritmias más sofisticadas.

Como última necesidad ponemos la de aprender sobre las tecnologías implicadas en el proyecto con el objetivo de tener varias vías con las que enseñar tecnología a todo aquel que quiera adentrarse en este mundo, desde la escuela hasta la investigación en universidades.

Este trabajo surge de la necesidad de obtener datos crudos para su posterior análisis automático. Por ello se ha descrito la arquitectura global de la que se servirán aplicaciones más abstractas pero en conclusión se ha comenzado a desarrollar desde las capas hardware más básicas.

Conceptos como Internet de las Cosas, Computación Distribuida, Computación Paralela, Hardware Abierto, Big Data, MashUps, Inteligencia Artificial, Fusión de Sensores o Análisis y Síntesis de Música, pueden ser extraídos del presente trabajo para desarrollos más profundos en el futuro.

Por tanto, la idea es clara, tener algo físico y real sobre lo que abstraer conceptos más avanzados en el mundo de las ciencias de la computación de la información justificando así nuestro trabajo previo.

1.3- Resumen

En este trabajo se ha tratado de integrar varias tecnologías para ilustrar las posibilidades de aplicaciones o servicios dentro del concepto de internet de las cosas.

Para ello se han puesto en marcha varias pruebas de concepto que dejan abiertas varias vías de desarrollo en función de las necesidades futuras y requisitos del demandante.

En concreto se ha puesto a prueba una arquitectura básica mediante dos miniordenadores, un microcontrolador con escudo ethernet y de audio, dos cámaras de vídeo, diodos LED, un semáforo LED, tres bombillas incandescentes, dos sensores de temperatura y humedad, 6 micrófonos electrect, una guitarra española, un cajón, una armónica, un mueble o armario a medida, un computador personal ATX, un servidor NAS, dos routers, tres altavoces auto amplificados y una conexión ADSL a Internet que nos permite comunicarnos con la plataforma de control y monitorización así como con otros usuarios de nuestro entorno, terceras partes y el resto de servidores de nuestra infraestructura.

Este trabajo no se caracteriza por obtener unos resultados determinados mediante una lógica de negocio, si no que trata de cimentar bien la base para poder después construir estructuras híbridas, con aplicaciones de todo tipo. Por lo tanto, estamos hablando de los cimientos sobre los que posteriormente se podrán establecer servicios o aplicaciones concretas mucho más avanzadas

de lo conseguido en el presente proyecto.

El resumen en un párrafo sería algo como construcción, investigación, desarrollo ágil e innovación en tecnologías del sonido, como instrumentos virtuales, para aplicaciones verticales de Inteligencia Artificial en el Internet de las Cosas.

2- Estado del Arte

2.1- Electrónica, Acústica y Electroacústica [ref. 7.1.2]

2.1.1- Electrónica

Como rama de conocimiento de la física tenemos la electrónica. Principalmente se diferencian los sistemas electrónicos analógicos de los digitales. Mientras en los sistemas analógicos se utilizan modelos continuos en los sistemas digitales los modelos son discretos.

Los componentes electrónicos básicos son las resistencias, los condensadores y las bobinas. También tenemos semiconductores como los diodos o los amplificadores operacionales. Con estos componentes se pueden crear puertas lógicas que son la base para la electrónica digital.

El principal proceso que vamos a tratar es la conversión analógico a digital (ADC) y su viceversa (DAC), ya que es el método con el que los computadores adquieren las señales de su entorno. Para ello, el primer paso es muestrear la señal a una frecuencia de reloj que cumpla ciertos requisitos respecto a la información que se tratará de computar. Seguidamente se termina de discretizar la señal pero esta vez en el eje de ordenadas, lo cuál se llama cuantificación, proceso que divide la señal en niveles representados por bits y en dónde mayor sea el número de niveles menor será el error o ruido de cuantificación introducido por el conversor pero también será mayor la cantidad de información o bits a codificar. Llegados al momento de codificar nos salimos de la parte de electrónica y nos adentramos en métodos numéricos capaces de minimizar o maximizar parámetros del flujo de bits a transmitir, almacenar o procesar.

Los filtros son circuitos electrónicos que actúan con su impedancia sobretodo en el dominio de la frecuencia. Pueden ser analógicos o digitales en DSP (Digital Signal Processor). Los tipos de filtro clásicos son el paso alto, paso bajo, paso banda y paso banda eliminada dónde el factor Q es el más importante. Pueden ser activos o pasivos, y de 1º, 2º o 3º orden. Específicamente tenemos el Butterworth, Chebyshev y Bessel, además de filtros de cruce como el de Linkwitz-Riley.

Los amplificadores son importantes dentro de un sistema de audio. Los podemos dividir en analógicos y digitales con su correspondientes clases. Los principales parámetros a tener en cuenta son la sensibilidad, la potencia de salida, el ancho de banda de potencia, la gama de frecuencia, la distorsión, la relación señal a ruido (SNR), la impedancia de entrada, la impedancia de fuente, el rechazo en modo común, el factor de amortiguación, la respuesta de fase y la polaridad.

Los equipos de procesamiento de señal o DSP, son dispositivos analógicos o digitales que adquieren la señal con un conversor, la procesan y la vuelven a convertir. Un ejemplo de estos dispositivos son las mesas de mezclas de sonido profesionales que incorporan subsistemas electrónicos para toda la etapa de tratamiento del audio.
Estos dispositivos normalmente son híbridos en cuanto a que pueden ser por hardware o virtuales, de los que destacan los basados en un sistema de reutilización de software de procesado, como los plug-ins.

Los dispositivos de audio más destacables son los ecualizadores, compresores,

expansores, limitadores, puertas de ruido y retardadores (delays) con los que se puede generar efectos como la reverberación artificial o el chorus.
Como último punto de un DSP para audio hay que resaltar que el sistema funciona en modo de tiempo real con el que se evitan latencias. Tiene un reloj común para las conversiones de señal y para los sincronismos del software digital que implementa las funciones digitales aplicables, siendo un sistema basado en interrupciones.

2.1.2- Acústica

Otra rama de la física es la acústica. Ésta se limita a tratar con ondas de presión que se transmiten o propagan por el medio físico (no por el vacío) y podemos llamar sonido. Por lo tanto, el sonido es una forma de energía mecánica con naturaleza ondulatoria y corpuscular ya que hablamos de ondas longitudinales que hacen vibrar las moléculas en torno a su centro de equilibrio.

El sonido lo podemos caracterizar por su frecuencia, su longitud de onda y por su velocidad. Además, tenemos el nivel de presión sonora que son pequeñas variaciones del nivel de presión barométrica y que se miden en dB con una escala logarítmica. Como con toda forma de energía en el sonido o acústica tenemos la potencia medida vatios por segundo. Otro parámetro importante de la acústica es el tipo de campo, que determina el modelo que se usa para tratar físicamente las ondas, dónde podemos tener campo libre o difuso así como campo cercano o lejano.

Una vez tenemos un modelo matemático que representa la señal sonora podemos diferenciar los tipos de sonido en tonos puros, señales complejas o ruido.

En la práctica, y en campos acústicos naturales, los tonos puros son sólo modelos ideales por lo que lo que prepondera son las señales complejas y el ruido. Ésta diferenciación es esencial ya que las señales complejas se pueden modelar con funciones trigonométricas mientras el ruido es un fenómeno puramente aleatorio o estadístico. Es importante tener claro el porqué de la escisión entre las matemáticas y la estadística ya que aunque en ambas se utilizan métodos numéricos para el desarrollo, el planteamiento de los problemas o condiciones iniciales, son muy diferentes conceptualmente, lo que cambia completamente la evolución de los modelos y por supuesto el resultado final para unos datos iniciales concretos.

Para calibrar un sistema de sonido acústicamente se utilizan estándares de medida, pero para configurarlo en un contexto determinado se hace necesaria la aplicación de métodos numéricos y simulaciones asistidas por computador que nos permiten visualizar todos los parámetros y adaptarlos a las condiciones acústicas. Por ejemplo, tenemos métodos de insonorización o aislamiento y de sonorización de recintos.

2.1.3- Electroacústica

Una vez tenemos un modelo mecánico del sonido, tenemos transductores que transforman esa energía de presión acústica en energía eléctrica o viceversa. Una vez la señal es eléctrica atiende a esta naturaleza simplificando todas las etapas internas entre el ADC y el DAC.

Por lo tanto, la electroacústica es la conjunción entre acústica y electrónica (o electricidad de bajo nivel), y se encarga de realizar dicha transducción con dispositivos electroacústicos llamados micrófonos (para percibir el sonido) y altavoces (para emitir sonido).

Los micrófonos, en cuanto a su membrana y cápsula, pueden ser de presión (omnidireccionales), de gradiente de presión (forma de ocho) o combinando ambos tipos.
Pero si queremos curvas o patrones polares más precisos en cuanto a direccionalidad, tenemos micrófonos del tipo cañón o de interferencia (muy direccionales a altas frecuencias y con patrones tipo trébol), de tipo zoom (directividad variable gracias a dos micrófonos en la misma cápsula), de tipo parabólico (respuesta en frecuencia dependiente del tamaño de la pantalla parabólica). Los micrófonos de superficie o de zona de presión son los utilizados habitualmente para realizar grabaciones o amplificaciones.

En cuanto al principio de conversión a electricidad tenemos dos grandes grupos, lo dinámicos (cinta y bobina móvil o electrodinámico) y los de condensador (LF(condensador estándar) , RF(radiofrecuencia) y Electrect).

Los altavoces suelen ser electrodinámicos (de bobina móvil) pero también pueden ser electroestáticos (de condensador). Una vez tenemos construida la parte transductora (la más compleja) podemos empotrar éste sistema transductor en una pantalla infinita (impedancia acústica infinita), o en una caja acústica con propiedades acústicas particulares que atienden a todos los detalles de su diseño y construcción modificando así la respuesta en frecuencia del altavoz. Además, tenemos los altavoces de tipo bocina caracterizados por su eficiencia y rendimiento en términos acústicos.

En un sistema de emisión de sonido profesional se mezclarán distintos tipos de altavoces separados con sus correspondientes filtros de cruce el el dominio eléctrico y ordenados en el espacio de acuerdo a los objetivos acústicos del diseño de sistema de sonido.

Las principales propiedades electroacústicas son el patrón de presión sonora (respuesta en frecuencia), el patrón de potencia acústica, la potencia eléctrica, la eficiencia en la transducción, las impedancias, el ruido interno y la sensibilidad. Y como parámetro estrella tenemos la directividad, representada con curvas polares al igual que otros sistemas radiantes como las antenas.

Cuando se trata de sistemas profesionales todos los conceptos se juntan, complicando el análisis y diseño, pero con los equipos y con los conocimientos científicos actuales éstos sistemas híbridos en cuanto a tipo son una ventaja. Y es que, con un buen diseño se pueden controlar todos los parámetros del sistema formando un sistema complejo que en su régimen de funcionamiento responde adecuadamente al diseño teórico. Como ejemplo de éstos sistemas híbridos tenemos desde altavoces de varias vías, en columna, en array o grupo así como con micrófonos, salas sonorizadas o recintos abiertos y al fin y al cabo todos los elementos acústicos y electrónicos necesarios para la aplicación electroacústica concreta de la que trate el diseño. También podemos tener varios canales de señal de sonido formando un espacio psicoacústico de 3 dimensiones y/o aproximaciones.

2.2- Interfaces Hombre-Máquina Audio-Visuales

Las interfaces hombre-máquina son un sector del conocimiento complejo. Las hay multimodales de forma que el usuario y el sistema interactúan a través de varias entradas y salidas.

En este trabajo, nos vamos a centrar en interfaces audiovisuales, concretamente en sistemas de habla y escucha, aunque estaremos apoyados por un sistema de vídeo y detección de movimientos.

A continuación mostraremos la teoría de un sistema de habla.

2.2.1- Sistemas de Habla

2.2.1.1-Panorámica

Las tecnologías del habla proporcionan un acceso natural a las fuentes de información. Así, son un medio natural con el que interactuar con las máquinas. Una de los aprendizajes mas importantes que se produce al ser niño es el hecho de hablar por lo que cuando llegue el momento en el que las interfaces hombre-máquina puedan usar como medio el habla empezarán su niñez.

Los avances en tecnologías de la información y las comunicaciones han hecho posible avanzar mucho en este campo haciendo posible resolver muchos problemas. Pero el problema del lenguaje natural en todos sus aspectos sigue siendo un problema complejo a resolver, es un bio-problema.

Existen muchas aplicaciones que usan o pudieran usar las tecnologías del habla, entre otras podemos enumerar: gestión del correo electrónico, gestión de consultas, dictado automático, indexado y navegación, verificación de identidad, análisis de sentimientos, análisis del fraude, comando y control para manos libres, atención automática al cliente, resúmenes automáticos y en fin: búsqueda, etiquetado, navegación, control, traducción y minería de la información basada en registros de voz.

2.2.1.2-Producción y Percepción del Habla

A- Producción de Voz

Reduciendo a términos físicos podemos llamar al habla voz. Así la voz es un fenómeno acústico originado por estructuras anatómicas. Los sistemas computacionales tratan de imitar estos procesos acústicos originados en el sistema vocal humano y que se suele llamar síntesis de voz.

Anatómicamente tenemos: cavidades (infraglóticas y supraglóticas), órganos fonadores (cuerdas vocales), órganos articulatorios (lengua, labios , etc..) y coadyuvantes (epiglotis y diafragma).

El funcionamiento se basa en la producción y moldeado de flujos de aire. El aire fluye de las zonas de baja presión a las de alta presión. Como principales corrientes tenemos la inspiración y la espiración.

El principal proceso implicado es la fonación, la cual genera el tono glotal. El tono glotal acústicamente es una onda de presión compleja con una frecuencia fundamental y unos armónicos asociados. Podemos diferenciar entre la presencia de fonación (sonoro, espirada, rota, falsetto y murmurada) o la ausencia de fonación (sordo y susurrado). Dependiendo de todo lo anterior tenemos una fuente compleja de tonos idealmente puros y de ruidos turbulentos.

A efectos de modelar el problema es muy importante tener en cuenta la resonancia de las ondas acústicas ya que determinan los llamados formantes. Gracias a éstos podemos diferenciar los sonidos de las vocales de los de las consonantes (sordas y sonoras).

Pero, para parametrizar las señales de voz y poder llegar a reconocer e identificar unívocamente a un locutor, o sea obtener su huella biométrica, hay que tener clara la diferencia entre las características acústicas y las fonéticas.

La acústica se refiere a las características intrínsecas del sonido como la periodicidad o a periodicidad, frecuencia fundamentales, espectros, etc...

La fonética se refiere a los parámetros relacionados con la percepción del sonido del receptor del habla como el tono, la intensidad o el timbre.

Con este resumen, espero que se haya entendido la complejidad de modelar la producción de habla analíticamente. Por eso, aunque cada vez menos, es fácil diferenciar cuándo nos habla una máquina y cuándo un humano. De hecho vamos a mencionar dos fenómenos muy importantes para la naturalidad del habla y reconocimiento del locutor; éstos son la prosodia (entonación, duración e intensidad) y la coarticulación.

Si la producción de voz es un bio-problema complejo hay que tener en cuenta que en la cadena del habla también entra en juego la percepción del oyente, problema probablemente más complejo.

B- Percepción de Voz

Personalmente opino que el problema de percepción es más complejo porque considero que la anatomía implicada en el fenómeno es mas “fina”. Además, el papel de la neuronas sensitivas tiene mucha importancia. Pero puesto que nos vamos a centrar en fenómenos de la física clásica vamos a llegar a entender mucho más de lo que se creería a priori sobre el aparato anatómico asociado a la percepción de sonido, el oído.

El oído se suele dividir en tres partes: externo (oreja y conducto auditivo), medio (tímpano, cadena osicular, músculos y trompa de Eustaquio) e interno (canales semicirculares, cóclea y nervio auditivo).

En cuanto a fisiología, también dividimos las partes del oído como en el caso anatómico. Por

funcionalidad la parte externa se encarga de captar y filtrar las ondas, la parte media realiza una adaptación de presiones entre el medio exterior y el de naturaleza líquida del interior y finalmente, en la parte interna se produce una transducción de información o energía cinética (presión) a impulsos nerviosos (electroquímica).
El análisis fisiológico de éste aparato receptor tiene mucha complejidad así que no vamos a profundizar ya que no es un problema directamente asociado a nuestro trabajo.

2.2.1.3-Análisis de la Señal de Voz

La señal de voz es aquella señal compleja que representa el habla. Podemos ver el habla como un problema de comunicación en el que hay un emisor, un canal y un receptor. Ya hemos visto las bases del emisor y del receptor humanos. Así el canal será nuestro sistema variable pero que no analizaremos pues queremos independizarlo de la propia señal.

Para computar el habla o físicamente la voz, es apropiado analizar cómo es el sistema en la forma humana o no computada. De esta manera nos será más inmediato construir interfaces hombre-máquina.

Así los humanos codificamos las ideas en palabras y frases como forma apropiada para después procesarlas y recuperarlas mentalmente. A día de hoy, hemos creado un sofisticado sistema de racionalización que utiliza como herramienta el lenguaje, siendo el verbal el más complejo. La civilización moderna, con la extensión del alfabetismo, tiende a considerar el lenguaje prioritariamente en su forma escrita, sin embargo hay muchos significados que únicamente pueden ser expresados de forma oral. Esto nos demuestra que si ya es complejo el lenguaje escrito de cara a un análisis, la forma oral tiene aún más complejidades intrínsecas.

Pues nuestro problema es la forma oral del lenguaje, vamos a tratar este como una señal acústica. Es sabido que la señal acústica es continua en el tiempo, sin embargo las unidades lingüísticas que representan el lenguaje son discretas. Por ésta diferencia se nos hace difícil determinar la relación entre los instantes temporales y las palabras que representan. Los conceptos que representan esta diferencia son el fonema, los alófonos y la coarticulación, sin olvidar las características prosódicas.
Además, en comunicaciones directas se añaden otros factores debidos al contexto como la comunicación no verbal o la localización de la fuente de sonido.
Todo esto nos hace ver cómo de complejo es el problema, y lo lejos que aún estamos de que las máquinas lleguen a emular conversaciones naturales.

Aún así, dentro de lo posible, se han conseguido ya admirables avances que consiguen sintetizar voz, analizarla e incluso emular sistemas de diálogo.

De momento, en este apartado vamos a ver como se puede analizar la señal de voz.

A-Adquisición

Lo primero a tener en cuenta para analizar la voz es conocer su naturaleza. Como ya hemos visto ésta es en forma de ondas de presión acústicas, de sonido. El nivel de sonido se puede medir en forma de intensidad sonora (SIL) o de presión sonora (SPL) y se expresa en decibelios.

Para adquirir el sonido en el receptor o reproducirlo en el emisor, se necesitan transductores que transformen la energía de acústica a mecánica y al fin en eléctrica, o viceversa. La señal eléctrica es la que se utiliza para analizar voz pues hace mucho más sencillo el transporte, almacenamiento y procesamiento de la misma.

La naturaleza de la señal eléctrica es analógica pero debido a las múltiples ventajas de los sistemas digitales vamos a convertirla. Para convertirla usaremos un conversor Analógico-Digital en el que se producen los procesos de muestreo y cuantificación de la señal para posteriormente codificarla en binario e incluso comprimirla en su caso.

B- Preprocesado

Una vez tenemos la señal en forma digital nos será mucho más sencillo y eficiente tratarla. Las primeras y principales características a tener en cuenta sobre la señal serán la frecuencia máxima y ancho de banda, y la distorsión introducida por el cuantificador y su SNR(Relación Señal a Ruido) asociada. Teniendo al cuantificador como elemento más determinante del proceso, llegamos a que la cuantificación logarítmica es la más apropiada para la naturaleza de la señal de voz ya que minimiza el error y por lo tanto la SNR, haciéndola además independiente de la señal. Pero, la cuantificación óptima sería una no uniforme. En cuanto a la codificación podemos tener muchos esquemas que influirán directamente en la capacidad de analizar la señal pero que veremos en detalle más adelante.

En cualquier caso, estos problemas ya están más que resueltos y dependiendo del problema en su conjunto será mejor aplicar una u otra estrategia.

C- Técnicas de Análisis Localizado

La motivación del análisis localizado es que la señal de voz es quasiestacionaria, por lo que sus propiedades estadísticas permanecen constantes en periodos cortos de tiempo.

Hablaremos de ventanas de tiempo con las diferentes tramas que se han de escoger según la estadística del aparato fonador humano. Se ha de encontrar un equilibrio entre el tamaño de ventana y abarcar varios periodos de las señales periódicas.

La tasa de actualización mínima hace que normalmente las ventanas adyacentes se solapen un 50%.

Los tipos de ventanas más usados son: Rectangular, Hamming, Hanning, Bartlett, Kaiser y Blackman. Cada cuál con su espectro y sus correspondientes propiedades espectrales que hacen unas más adecuadas que otras dependiendo del uso.

D- Técnicas de Análisis en el Dominio del Tiempo

En el tiempo se procesa directamente la forma de onda de una o más señales de forma muy eficiente. Para computar, se utiliza una transformación no lineal de la que lo que más nos importa es la energía, la amplitud, los cruces de la señal por cero y su función de auto-correlación, y es a partir de estos parámetros en los que se basan todas las técnicas más sofisticadas de tratamiento de señal en el tiempo.

E-Técnicas de Análisis en el Dominio de la Frecuencia

En frecuencia podemos hacer análisis más consistentes que en el tiempo con métodos como utilizar bancos de filtros o transformadas locales como la de Fourier. También tenemos la transformada discreta de Fourier presumiblemente más rápida al computar.

El espectro de la señal es todo lo que tenemos y éste puede ser de banda estrecha o banda ancha. Podemos representar el espectrograma en cascada y en escala de grises dónde veremos todos los detalles de la señal en frecuencia.

F- Extracción de Características

Para poder llegar a identificar a un locutor hay que utilizar técnicas de extracción de características de la señal en todos sus aspectos. Con estas características se entrenará un clasificador y se podrá decidir si un locutor es quién dice ser ya que éste estará representado por un patrón reconocible por el clasificador.
Es posible desvelar el comportamiento de un sistema en cuanto éste sea predecible. Si nos enfrentamos a patrones no lineales o caóticos no predecibles, a priori, no tendríamos posibilidades de controlarlo.

Una técnica es la predicción lineal, que permite modelar el mecanismo de producción de voz del hablante y extraer únicamente las características relevantes de la voz. En la estimación de los coeficientes del predictor correctamente radica la calidad del sistema ya que éste tendrá menos error y ruido.

Otra técnica es el análisis cepstral que analiza la envolvente de la señal ya que contiene información relevante. En el dominio cepstral las componentes se combinan linealmente y son independientes, requisitos adecuados para su posterior análisis. Quefrencia: Tiempo y frecuencia invierten sus papeles.

La Estimación de la frecuencia fundamental puede realizarse en el tiempo, en la frecuencia y en la quefrencia: parámetro esencial para caracterizar un locutor.

2.2.1.4-Codificación de Voz

Una vez hemos muestreado y cuantificado la señal hay que codificarla con un codificador adecuado a la aplicación. Con la codificación eliminaremos redundancias de manera que perceptualmente sea aceptable.

Para diseñar un buen codificador atenderemos principalmente al aparato fonador y auditivo humano. Los principales requisitos a cumplir por el codificador son una buena calidad, el retardo de codificación, robustez, complejidad y coste, codificaciones en cadena y transcodificación así como la transmisión de datos en la banda de la voz. Como en todos los ámbitos este proceso ya está estandarizado por organismo internacionales que principalmente han desarrollado escalas objetivas y subjetivas para evaluar la calidad del codificador.

Los tipos de codificadores se pueden dividir en codificadores de forma de onda, codificadores de voz o vocoders y codificadores híbridos. En función de la aplicación a la que se dediquen se utilizará un estándar u otro para codificar la señal.

2.2.1.5-Síntesis de Voz

Como todo el proceso de tratamiento de voz la síntesis de voz es un problema ya resuelto por lo que no nos pararemos a ver los detalles.

Simplemente se trata de emular un aparato fonador humano de la manera más cercana al caso físico real y reproducir el sonido por altavoces.

2.2.1.6-Reconocimiento de Habla

El reconocimiento de habla es más un problema de inteligencia artificial o percepción computacional que de tratamiento de señal, pero este proceso depende mucho del anterior análisis de la señal. Más concretamente es un problema de reconocimiento de patrones, dónde el objetivo es encontrar la secuencia de palabras más probable dada una secuencia de observaciones acústicas.

Antes de empezar debemos acotar el problema adaptándolo a la aplicación en la que se vaya a usar. Debemos plantearnos según nuestra aplicación lo siguiente: la dependencia o independencia del locutor, palabras aisladas o conectadas, el tamaño del vocabulario, el grado de confusión del vocabulario, el entorno de operación y la información lingüística disponible y sus restricciones.

Una vez hemos acotado nuestro problema se puede formalizar con una MAP (Máximo a Posteriori) o reformularlo adaptándolo a las observaciones acústicas mediante un ML o estimador de máxima verosimilitud. A partir de aquí hemos de usar el teorema de Bayes junto con el

modelo acústico y el modelo del lenguaje.

Lo primero que necesitamos una vez planteado bien el problema es un extractor de características veraz.

Lo podemos tener basado en el modelo de producción mediante predicción lineal, basado en el modelo de percepción o auditivo con coeficientes cepstrales escalados MEL(MFCC) o predicción lineal perceptual. También siempre podemos basarnos en ambos y hacer soluciones híbridas con parámetros dinámicos.

Para el modelado acústico podemos utilizar DTW (Dynamic Time Warping) u otras alternativas como los modelos ocultos de Markov(HMM) o las redes neuronales no recurrentes.
Para el modelado lingüístico se utilizan modelos estocásticos que ayudan a las decisiones de los HMM.

2.2.1.7-Sistemas de Diálogo

Los sistemas de diálogo automático se utilizan cuando la información a transmitir y la interacción con el usuario es demasiado compleja para hacerlo por otras vías que no sean el habla.

Los componentes tecnológicos que implica un sistema de diálogo son un reconocedor de habla, un procesador lingüístico, un gestor de diálogo, una interfaz de acceso a la base de datos, un generador de lenguaje y un conversor texto-voz.

El componente más esencial y complejo en un sistema de diálogo es el propio gestor del diálogo. Éste representa la inteligencia del sistema ya que no entiende palabras sino estructuras de datos que representan conceptos. Su objetivo es satisfacer al usuario cooperando con él. Así, se puede estructurar en apertura, cuerpo y cierre.

Podemos establecer modelos para el gestor basados en aproximaciones; estructurales, orientadas a la planificación o de interacción racional. Visto de otro modo son respectivamente basadas en diagramas de estados finitos y gramáticas formales, basadas en objetivos y planes o basadas en comportamientos racionales dinámicos representados mediante formalismos lógicos.

2.2.1.8-Reconocimiento de Locutores

El reconocimiento de locutores se puede ver como un problema biométrico. La biometría ofrece múltiples ventajas para la identificación y verificación de entidades humanas respecto al clásico de usuario y contraseña.

Un sistema biométrico basado en locutor está compuesto principalmente por un módulo de reconocimiento de locutor, uno de reconocimiento de habla y otro de análisis y decisión que da los resultados obtenidos. Por supuesto también se dispone de sensores y actuadores así como bases de datos.

Se ha de diferenciar entre sistemas de verificación de los de identificación. La

verificación es un problema de decisión entre 2 clases, mientras que la identificación es un problema de N clases.

La evaluación de estos sistemas responde a criterios formales basados matrices de verdadero y falso y representados habitualmente mediante curvas ROC. Los costes de detección son otro parámetro muy importante a tener en cuenta

2.3- Inteligencia Artificial y Robótica

La Inteligencia Artificial (IA) y la Robótica son campos de conocimiento muy amplios e incluso ambiguos ya que atienden a múltiples definiciones según la especialidad científica que los analice, investigue y desarrolle. Personalmente creo que la especialidad que mejor los define es la filosofía de la ciencia ya que ésta abarca desde las raíces teóricas hasta las aplicaciones en ciencia. Debido a que estamos en un contexto de ingeniería los conceptos están adaptados a las aplicaciones que surgen tanto de las teorías formales como de las basadas en la experimentación. Por todo esto, hemos decidido exponer el contenido de la enciclopedia más aceptada en Internet, la Wikipedia.

*2.3.1- Inteligencia Artificial: Categorización e Historia * [ref. 7.1.1]*

2.3.1.1- Categorización

*"**Búsqueda heurística:** Podemos definir una heurística como un truco o estrategia que limita grandiosamente la búsqueda de soluciones ante grandes espacios de problemas. Por lo tanto ante un problema, nos ayuda a seleccionar las bifurcaciones, dentro de un árbol, con más posibilidades, con ello se restringe la búsqueda aunque no siempre se garantiza una solución adecuada. Todo lo que se debe tener para que una heurística sea adecuada es que nos proporcione soluciones que sean lo suficientemente buenas. Además utilizando la heurística, no será necesario replantear un problema cada vez que se afronte, ya que si lo hemos planteado anteriormente, ésta sugerirá la forma en que se ha de proceder para resolverlo."*

*"**Representación del conocimiento:** La representación es una cuestión clave a la hora de encontrar soluciones a los problemas planteados, y que además éstas sean adecuadas. Si analizamos más detenidamente el término y además dentro de la Informática, y más concretamente dentro de la Inteligencia Artificial, encontramos varias definiciones". "El tema central de esta disciplina es el estudio del conocimiento y su manejo."*

"Ha sido caracterizado por el intento continuo de conseguir más y mejores estructuras de representación del conocimiento, junto con técnicas adecuadas para su manipulación,

que permitiesen la resolución inteligente de algunos de los problemas ya planteados. Otra característica a resaltar, es la inclusión en los programas de inteligencia artificial, aunque por separado, de los conocimientos y la unidad que controla y dirige la búsqueda de soluciones. Dada esta disposición, en estos programas se hace fácil la modificación, ampliación y actualización de los mismos. Hemos de tener en cuenta que un programa de Inteligencia Artificial, tiene toda la información interconectada e interrelacionada, y estas interconexiones se utilizaran para representar relaciones unidas al conocimiento genérico sobre el problema planteado. Para que un problema sea tratado adecuadamente, sabiendo que no es fácil la representación del conocimiento en un entorno concreto, hemos de elegir un esquema de representación acorde con la naturaleza del dominio de conocimiento donde se vaya a trabajar. "

*"**Lenguajes, entornos y herramientas de Inteligencia Artificial:** En la Inteligencia Artificial, se han desarrollado diferentes lenguajes específicos para los diferentes campos de aplicación. Estos lenguajes en su mayoría cuentan con una serie de características comunes que podemos resumir de la siguiente forma: Este tipo de software ofrece una gran modularidad. Poseen gran capacidad de tomar decisiones de programación hasta el último momento, es decir cuando el programa ya está ejecutándose. Ofrecen grandes facilidades en el manejo de listas, y esto es importante, ya que las listas son la estructura más habitual usada para la representación del conocimiento en la Inteligencia Artificial. Facilitan la realización de ciertos tipos de deducción automática permitiendo también, la creación de una base de hechos, que es el lugar donde se recogen los datos iniciales del problema a resolver y también los resultados intermedios una vez obtenidos. Permite el uso simultáneo de estructuras que incorporan conocimiento declarativo y conocimiento procedimental. Tienen una marcada orientación gráfica. Además, las herramientas de Inteligencia artificial, permiten hacer un seguimiento de todos los cambios realizados a lo largo de toda la sesión. Disponen de herramientas capaces de desarrollar programas que son capaces de comprender otros programas y también de realizar modificaciones sobre ellos."*

*Nota: Los * significan que el apartado titulado así ha sido copiado directamente respetando los derechos de uso y difusión de Wikipedia.*

Stuart Russell y Peter Norvig diferencian estos tipos de la inteligencia artificial:

*-"**Sistemas que piensan como humanos:** Estos sistemas tratan de emular el pensamiento humano; por ejemplo las redes neuronales artificiales. La automatización de actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, la resolución de problemas y el aprendizaje."*

***-"Sistemas que actúan como humanos:** Estos sistemas tratan de actuar como humanos; es decir, imitan el comportamiento humano; por ejemplo la robótica. El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor."*

*-"**Sistemas que piensan racionalmente:** Es decir, con lógica (idealmente), tratan de imitar o emular el pensamiento lógico racional del ser humano; por ejemplo los sistemas expertos. El estudio de los cálculos que hacen posible percibir, razonar y actuar."*

*-"**Sistemas que actúan racionalmente (idealmente):** Tratan de emular de forma racional el comportamiento humano; por ejemplo los agentes inteligentes. Está relacionado con conductas inteligentes en artefactos."*

Escuelas de pensamiento:

La IA se divide en dos escuelas de pensamiento:

-La inteligencia artificial convencional
-La inteligencia computacional

A-Inteligencia artificial convencional

Se conoce también como IA simbólico-deductiva. Está basada en el análisis formal y estadístico del comportamiento humano ante diferentes problemas:

***-Razonamiento basado en casos:** Ayuda a tomar decisiones mientras se resuelven ciertos problemas concretos y, aparte de que son muy importantes, requieren de un buen funcionamiento.*

***-Sistemas expertos:** Infieren una solución a través del conocimiento previo del contexto en que se aplica y ocupa de ciertas reglas o relaciones.*

***-Redes Bayesianas:** Propone soluciones mediante inferencia probabilística.*

***-Inteligencia artificial basada en comportamientos**: Esta inteligencia contiene autonomía y puede auto-regularse y controlarse para mejorar.*

***-Smart process management:** Facilita la toma de decisiones complejas, proponiendo una solución a un determinado problema al igual que lo haría un especialista en la dicha actividad.*

B-Inteligencia artificial computacional

La Inteligencia Computacional (también conocida como IA subsimbólica-inductiva) implica desarrollo o aprendizaje interactivo (por ejemplo, modificaciones interactivas de los parámetros en sistemas conexionistas). El aprendizaje se realiza basándose en datos empíricos.

2.3.1.2- Historia [ref. 7.1.2.2]*

-El término "inteligencia artificial" fue acuñado formalmente en 1956 durante la conferencia de Darthmounth, más para entonces ya se había estado trabajando en ello durante cinco años en los cuales se había propuesto muchas definiciones distintas que en ningún caso habían logrado ser aceptadas totalmente por la comunidad investigadora. La IA es una de las disciplinas más nuevas junto con la genética moderna. Además la historia contextualizada de la Inteligencia Artificial es muy dependiente del objeto del análisis.

2.3.2- Cibernética: Ingeniería de Control de Sistemas [ref. 7.1.1.2]*

El estudio interdisciplinario de la estructura de los sistemas reguladores es la cibernética. Está íntimamente ligada a la teoría de control y a la de sistemas. En especial se utiliza la teoría de control mediante realimentaciones del sistema y subsistemas.

Es una ciencia que nace en 1942 y que impulsa la inteligencia artificial durante los 50 y la teoría de la información en la década de los 60.

"La cibernética, según el epistemólogo, antropólogo, cibernetista y padre de la terapia familiar, Gregory Bateson, es la rama de las matemáticas que se encarga de los problemas de control, recursividad e información. Bateson también afirma que la cibernética es "el más grande mordisco a la fruta del árbol del Conocimiento que la humanidad haya dado en los últimos 2000 años"."

"Stafford Beer, filósofo de la teoría organizacional y gerencial, de quien el propio Wiener dijo que debía ser considerado como el padre de la cibernética de gestión, define a la cibernética como "la ciencia de la organización efectiva"."

"Según el Profesor Dr. Stafford Beer, la cibernética estudia los flujos de información que rodean un sistema, y la forma en que esta información es usada por el sistema como un valor que le permite controlarse a sí mismo: ocurre tanto para sistemas animados como inanimados indiferentemente. La cibernética es una ciencia interdisciplinar, y está tan ligada a la física como al estudio del cerebro como al estudio de los computadores, y tiene también mucho que ver con los

lenguajes formales de la ciencia, proporcionando herramientas con las cuales describir de manera objetiva el comportamiento de todos estos sistemas."

Mucha gente asocia la cibernética con la robótica, los robots y el concepto de cyborg debido al uso que se le ha dado en algunas obras de ciencia ficción, pero desde un punto de vista estrictamente científico, la cibernética trata acerca de sistemas de control basados en la retroalimentación.

Tanto la ingeniería de sistemas como la de control, o sea la cibernética, está muy relacionada con la inteligencia artificial que a continuación veremos.

2.3.3- Inteligencia Artificial: Técnicas y Campos

[ref. 7.1.3][ref. 7.1.4]

La Inteligencia Artificial (IA) es un vasto campo de conocimiento científico en el que hay muchas maneras de categorizar cada especialidad como ya hemos visto la de Wikipedia. Ahora vamos a hacer otra clasificación que engloba tanto técnicas como campos de aplicación de la IA:

- **Aprendizaje Automático:** El objetivo es desarrollar técnicas que permitan a los computadores aprender. Tiene una inmensidad de aplicaciones sobretodo si se junta con otras técnicas como de representación del conocimiento.

- **Minería de Datos:** Es el conjunto de técnicas y tecnologías que permiten explorar grandes bases de datos (Big Data), de manera automática o semi-automática paara encontrar patrones, tendencias y reglas que expliquen el comportamiento de los datos en un determinado conteto.

- **Ingeniería del Conocimiento:** Su objetivo es extraer, articular e informatizar el conocimiento de un experto, dándo lugar a los sistemas expertos.

- **Redes Bayesianas:** Su objetivo es representar el conocimiento y el razonamiento probabilístico y multivariable de una forma gráfica, con grafos dirigidos.

- **Técnicas de Representación del Conocimiento:** Se persigue representar, con predicados, el conocimiento para facilitar la inferencia formal. Se aplican símbolos del dominio del discurso, operadores modales y cualquier lógica que ofrezca una semántica formal de como las funciones de razonamiento se aplican.

-**Visión Artificial:** Es aquel campo que estudia la percepción computacional a través de cámaras y que busca imitar las capacidades de visión y percepción humana y animal.

- **Lingüística Computacional:** Realiza aplicaciones informáticas que imitan la capacidad humana de hablar y entender. Por ejemplo, tenemos traductores automáticos,

reconocedores de habla y cualquier aplicación en la que haya texto o lenguaje humano.

- **Procesamiento del Lenguaje Natural:** Es más una especialidad de las interfaces de usuario que aplica técnicas de IA para entender y procesar el habla o lenguaje natural humano.

- **Vida Artificial:** Es el estudio de la vida y de los sistemas artificiales que exhiben propiedades similares a los seres vivos, mediante modelos de simulación. Pertenece al campo de las ciencias de las complejidad, que son sistemas que tienen propiedades emergentes como conjunto diferentes a las propiedades individuales de cada elemento. Se pueden estudiar ecosistemas, economías, culturas y sistemas físicos no lineales o caóticos como el cerebro, el sistema nervioso, las turbulencias, la geología o el tiempo atmosférico.

2.3.4- Robótica [ref. 7.1.1.2]*

*"La **robótica** es la rama de la tecnología que se dedica al diseño, construcción, operación, disposición estructural, manufactura y aplicación de los robots."*

"La robótica combina diversas disciplinas como son: la mecánica, la electrónica, la informática, la inteligencia artificial, la ingeniería de control y la física.3 Otras áreas importantes en robótica son el álgebra, los autómatas programables, la animatrónica y las máquinas de estados."

La estructura, es definida por el tipo de configuración general del Robot, puede ser metamórfica. El concepto de metamorfismo, de reciente aparición, se ha introducido para incrementar la flexibilidad funcional de un Robot a través del cambio de su configuración por el propio Robot. El metamorfismo admite diversos niveles, desde los más elementales (cambio de herramienta o de efecto terminal), hasta los más complejos como el cambio o alteración de algunos de sus elementos o subsistemas estructurales. Los dispositivos y mecanismos que pueden agruparse bajo la denominación genérica del Robot, tal como se ha indicado, son muy diversos y es por tanto difícil establecer una clasificación coherente de los mismos que resista un análisis crítico y riguroso. La subdivisión de los Robots, con base en su arquitectura, se hace en los siguientes grupos: poliarticulados, móviles, androides, zoomórficos e híbridos.

Según la estructura del robot construido se puede diferenciar las siguientes clases:

-1. Poliarticulados

En este grupo se encuentran los Robots de muy diversa forma y configuración, cuya característica común es la de ser básicamente sedentarios (aunque excepcionalmente pueden ser guiados para efectuar desplazamientos limitados) y estar estructurados para mover sus elementos terminales en un determinado espacio de trabajo según uno o más sistemas de coordenadas, y con un número limitado de grados de libertad. En este grupo, se encuentran los manipuladores, los Robots

industriales, los Robots cartesianos y se emplean cuando es preciso abarcar una zona de trabajo relativamente amplia o alargada, actuar sobre objetos con un plano de simetría vertical o reducir el espacio ocupado en el suelo.

-2. Móviles

Son Robots con gran capacidad de desplazamiento, basados en carros o plataformas y dotados de un sistema locomotor de tipo rodante. Siguen su camino por telemando o guiándose por la información recibida de su entorno a través de sus sensores. Estos Robots aseguran el transporte de piezas de un punto a otro de una cadena de fabricación. Guiados mediante pistas materializadas a través de la radiación electromagnética de circuitos empotrados en el suelo, o a través de bandas detectadas fotoeléctricamente, pueden incluso llegar a sortear obstáculos y están dotados de un nivel relativamente elevado de inteligencia.

*Nota: Los * significan que el apartado titulado así ha sido copiado directamente respetando los derechos de uso y difusión de Wikipedia.*

-3. Androides o Humanoides

Son Robots que intentan reproducir total o parcialmente la forma y el comportamiento cinemático del ser humano. Actualmente, los androides son todavía dispositivos muy poco evolucionados y sin utilidad práctica, y destinados, fundamentalmente, al estudio y experimentación. Uno de los aspectos más complejos de estos Robots, y sobre el que se centra la mayoría de los trabajos, es el de la locomoción bípeda. En este caso, el principal problema es controlar dinámica y coordinadamente en el tiempo real el proceso y mantener simultáneamente el equilibrio del Robot.

-4. Zoomórficos

Los Robots zoomórficos, que considerados en sentido no restrictivo podrían incluir también a los androides, constituyen una clase caracterizada principalmente por sus sistemas de locomoción que imitan a los diversos seres vivos. A pesar de la disparidad morfológica de sus posibles sistemas de locomoción es conveniente agrupar a los Robots zoomórficos en dos categorías principales: caminadores y no caminadores. El grupo de los Robots zoomórficos no caminadores está muy poco evolucionado. Los experimentos efectuados en Japón basados en segmentos cilíndricos biselados acoplados axialmente entre sí y dotados de un movimiento relativo de rotación. Los Robots zoomórficos caminadores multípedos son muy numerosos y están siendo objeto de experimentos en diversos laboratorios con vistas al desarrollo posterior de verdaderos vehículos terrenos, pilotados o autónomos, capaces de evolucionar en superficies muy accidentadas. Las aplicaciones de estos Robots serán interesantes en el campo de la exploración espacial y en el estudio de los volcanes.

-5. Híbridos

Corresponden a aquellos de difícil clasificación, cuya estructura se sitúa en combinación con alguna de las anteriores ya expuestas, bien sea por conjunción o por yuxtaposición. Por ejemplo, un

dispositivo segmentado articulado y con ruedas, es al mismo tiempo, uno de los atributos de los Robots móviles y de los Robots zoomórficos.

2.3.5- Aplicaciones

[ref. 7.1.3] [ref. 7.1.4] [ref. 7.1.5] [ref. 7.1.6] [ref. 7.1.11] [ref. 7.1.10]

Hasta el momento, hemos visto definiciones, opiniones, clasificaciones e historia sobre la IA, la robótica y la cibernética. Para ello nos hemos apoyado totalmente en el conocimiento comunitario que ofrece la Wikipedia. En éste apartado de aplicaciones vamos a ver algunas que pueden ser aplicadas en nuestro sistema de forma breve pues corresponderían a líneas futuras del proyecto.

A- Inteligencia Ambiental y Inteligencia de Contexto

La inteligencia ambiental y de contexto trata de dar "sentido común" a los sistemas que la utilizan. Es un complejo campo muy dependiente de la aplicación en la que es usada. Pero aquí la mostramos ya que como base utiliza datos medidos de la realidad física para a partir de aquí abstraer conceptos y poder modelar comportamientos del sistema.

B- Fusión de Sensores y Actuadores

Para modelar comportamientos y ofrecer contexto a las aplicaciones primero se ha de fusionar e integrar todos los componentes que forman el sistema para que actúen al unísono ante eventos. Éstos eventos serán capturados por sensores que una vez fusionados podrán ofrecer respuestas inteligentes mediante actuadores físicos o virtuales.

C- Domótica

La domótica es otra rama que integra muchos de los conceptos vistos anteriormente. Se puede ver como la ciencia de hacer que las casas tengan cierta inteligencia para aportar valor a los residentes en cualquier sentido.

D- Hilos de Control de la Seguridad del Sistema

En este apartado nos referimos a la inteligencia aplicada a la seguridad de los sistemas de información. Un sistema completo de seguridad está formado por varios hilos de razonamiento intercomunicados entre ellos que actúan de forma preventiva o reactiva en cuánto a defensa, y pasiva o activa en cuanto a ofensiva.

E- Telecontrol Automático: Navegación y Guiado

También relacionado con nuestro trabajo tenemos los sistemas que controlan en remoto todos los detalles de la aplicación. Como ejemplo podemos ver cómo un dron es dirigido

a distancia mediante técnicas de navegación con planificación de rutas y con muchos otros tipos de comportamiento específicos de la aplicación.

F- Análisis y Síntesis de Audio y Música

Quizás la aplicación de técnicas clásicas para síntesis y análisis automático de registros acústicos no tenga demasiado interés industrial y comercial. Además, es un campo de conocimiento tan antiguo como lo es la música dónde las técnicas clásicas ya son bien conocidas.

Pero hay que tener en cuenta, que el interés investigador es muy amplio, debido por ejemplo, a las semejanzas entre el lenguaje musical y el verbal. Además, otra importante semejanza, es la de las técnicas de reconocimiento de movimiento y las técnicas de interés musical.

Como ya hemos dicho las técnicas clásicas ya han sido muy investigadas, pero las técnicas de inteligencia artificial bio-inspiradas tienen mucho interés científico actualmente ya que son aplicables a muchos ámbitos tanto científicos como artísticos.

Personalmente pienso que la música como medio y fin de investigación es muy interesante ya que tiene tanto aspectos estructurales o analíticos como artísticos o creativos, lo que creo que es un reto de investigación en el presente siglo. Además, tiene las ventajas de ser fácilmente enseñado en forma de música y hacer de la educación un espacio más auditivo y menos visual.

Por supuesto, también tenemos aplicaciones militares con el audio digital, con arrays de micrófonos para detectar explosiones, proyectiles y en fin ondas de presión acústica, ya se a el medio el aire, el agua o las vías de tren.

También hace de complemento perfecto de los actuales sistemas CCTV de videovigilancia, pudiendo detectar voces específicas de una persona e identificarlas con un patrón biométrico, gestionarlas y procesarlas por si ha sido algún comando u orden el sonido que se ha detectado.

En los estadios multitudinarios también se utilizan sistemas de microfonía para captar los gritos de los jugadores, público y detectar el balón como objetivo a perseguir o a hacer tracking o detectores de movimiento en vídeo para también perseguir al balón y a los jugadores así como drones para captar imágenes ambiente del estadio y de toda la multitud.

La última aplicación de la física que utiliza las mismas técnicas de funcionamiento es la acústica submarina con subsistemas electrónicos como los nuestros pero resistentes al agua o a cualquier otro líquido o medio en el que esté. Se trata de analizar o barrer el medio oceánico consiguiendo mapas de distancias hasta objetos, rocas, fallas, animales, plantas, submarinos artificiales, corrientes oceánicas, profundidad y un montón de parámetros fáciles de procesar posteriormente mediante el HPC RaspberryPi y ProceSiX y Thinger.

2.4- Internet de las Cosas, Seguridad y WebRTC

[ref. 7.1.9] [ref. 7.1.11] [ref. 7.1.10] [ref. 7.1.8] [ref. 7.1.6] [ref. 7.1.5]

2.4.1- Internet de las Cosas

Echando la vista atrás, vemos como hace pocos años se inventó HTML y CSS con su respectivo protocolo HTTP. Con esto tuvimos en un principio páginas web estáticas. A continuación, nacieron las páginas web dinámicas con sus lenguajes asociados como PHP.

Los métodos semánticos de HTTP son : GET, DELETE, POST, PUT , PATCH, LINK, UNLINK, HEAD, OPTIONS. ¿Qué métodos deberemos usar?

Después, bastante próximas en el tiempo nacieron las aplicaciones web y las aplicaciones móviles. En mi opinión las aplicaciones web no son mas que páginas web dinámicas más complejas o enriquecidas en diferentes aspectos. En cuanto a las aplicaciones móviles nacieron como otra burbuja para cubrir la necesidad de las mejoras introducidas tanto en las redes móviles como en los dispositivos móviles y así aprovechar su hardware nativo.

A día de hoy, ya es palpable que ambas burbujas se han unido para convertirse en aplicaciones totalmente multiplataforma, tendencia lógica que ya implantó Java con su máquina virtual.

Para hacer posible este fenómeno no quedaba mas que pasar por procesos de estandarización a las que las diferentes plataformas se tendrán que adaptar. Principalmente tenemos dos importantes evoluciones como son el HTML en la versión 5 y CSS en la versión 3. Además como lenguaje dinámico tenemos el estándar de ECMAScript que aproximadamente es cumplido por Javascript.

Con estas tres tecnologías se pueden desarrollar clientes multiplataforma independientemente del hardware en el que se ejecuten. La tendencia es que se abstraiga el sistema operativo siendo el entorno de ejecución de las aplicaciones un navegador web, pudiendo así funcionar las aplicaciones en teléfonos, televisiones, ordenadores, etc...Un ejemplo de esta tendencia hacia un entorno estándar, común y abierto y en resumen WEB, es FirefoxOS o Tizen.

Al ser las aplicaciones ubicuas, en muchos casos se necesita un lado servidor en la conexión, para el cual hay muchos lenguajes como Java, Ruby, Python o el mismo Javascript.

Como venimos viendo, la mejor estrategia es basarse en una pila que cumpla con los estándares en la medida de lo posible.

Puesto que la convergencia hace necesario entender bien los detalles de implementación, vamos a introducir algunos conceptos y tecnologías.

Tradicionalmente se han estructurado las aplicaciones web con una arquitectura cliente-servidor, dónde front-end será el cliente y back-end el servidor.

Habitualmente, en el cliente reside la interfaz de usuario (GUI) mientras en el servidor reside la base de datos o modelo de datos que ofrece persistencia, el propio servidor web que gestiona las peticiones de entrada y salida y el controlador o lógica de negocio dónde se implementa la funcionalidad de la aplicación.
Por popularidad, se ha implantado una arquitectura que representa lo expuesto anteriormente, este es el patrón de diseño MVC (Model-View-Controller) que puede tener distintas variaciones.

Actualmente se está implantando que cada entidad de este patrón se comunique con otro a través de servicios web. Los servicios web son un método de comunicación que lógicamente encapsula un mensaje codificado de alguna manera que hace factible que cualquier parte de una aplicación se pueda comunicar con otra de manera transparente. En internet todo está orientado al servicio. La tendencia actual es sustituir los tradicionales servicios web por el paradigma de los micro-servicios que consiguen arquitecturas aún mas eficientes, transparentes, ubicuas y al fin y al cabo adaptadas al crecimiento de Internet.

Debido a los tiempos que impone la red a las aplicaciones, otra tendencia actual es introducir la aplicación en su totalidad en el cliente, haciendo que todo resida en éste y dejando sólo para el servidor las tareas muy pesadas. Este tipo de aplicaciones se llaman SPA (Single Page Application) en las que si las tareas no son demasiado pesadas presumen de una interacción con el usuario mucho mas fluida, haciéndolas mas atractivas.

Dentro del desarrollo web, la tendencia es implementar aplicaciones en base componentes web con los lenguajes estándar o bien con híbridos para el móvil y la web..

En cuanto al desarrollo móvil, se pueden desarrollar aplicaciones nativas con los SDK (Software Development Kit) de cada sistema. Pero la tendencia es desarrollar aplicaciones híbridas que una vez implementadas se pueden desplegar en diferentes sistemas móviles o directamente en la web.

Tanto las tecnologías móviles como las web crecen a un ritmo vertiginoso y cualquier desarrollador dispone de un extenso maletín de herramientas que se mejoran día a día y en las que es necesario un continuo reciclamiento de conocimientos para mantener el ritmo.

El Internet de las Cosas (IoT) es un paradigma de computación en el cuál existen objetos que comparten información bidireccional vertical a todos los modelos de redes telemáticas. Coloquialmente, estos objetos son llamados cosas, pueden ser sensores industriales o domóticos, dispositivos electrónicos e informáticos y cualquier otra cosa imaginable.

Los servicios telemáticos avanzan a gran velocidad y existen muchas arquitecturas posibles para implementar un servicio a distancia. Un servicio para IoT es vertical respecto a las capas de los diferentes protocolos de comunicación usados en Internet habitualmente.

A día de hoy se usan servicios web orientados a objetos y al dato o la fuente, para ofrecer desde un servicio de tiempo real hasta integraciones entre dominios de empresas. Podemos usar diferentes técnicas como RPC con JSON o XML pero la tendencia es usar REST que puede tratar los datos directamente como objetos Javascript con JSON. También podemos enviar datos y multimedia mediante el estándar de los websockets.

Existen muchos tipos de datos en JSON como son los data types; de la clase Object, String, Number, Boolean, Null o Array. En cuanto a seguridad hay que tener cuidado con la seguridad en la notación javascript para no provocar involuntariamente Cross-Site Request Forgery (CSRF), ataques de injección de comandos, de Cross-Site Scripting (XSS) y sobre todo no generar agujeros de seguridad llevando mucha disciplina en el plan de desarrollo.

Con JavaScript XmlHttpRequest (XHR) y los Web APIs podemos contruir cualquier arquitectura orientada al recurso con JSON.

Los frameworks para construir aplicaciones ya sean MVC o SPA funcionan cogiendo recursos mediante el sistema JSON, por lo que el lado cliente consume y genera datos estrucutrados en formato JSON que serán procesados por el resto de la aplicación.

Una parte de ese resto de la aplicación sería el controlador y la vista pero no olvidemos la base de datos que puede ser orientado al dato o llamada NoSQL, por ejemplo CouchDB cumple los requisistos.

En el lado servidor hay que serializar, deserializar y responder mediante lenguaje JSON. Se puede hacer con cualquier lenguaje para la web, nuestro caso será javascript, python y java.

Si queremos hacer aplicaciones híbridas vamos a usar el framework Ionic, que está basado fuertemente en Angularjs en cliente web y en Cordova en cliente móvil. La clave es tener todas las partes desde el cliente hasta la base de datos pasando por el controlador y el middleware basadas en Javascript , CSS3 y HTML5.

En cuanto al paradigma de IoT, la red es Internet aunque esté compuesta por subredes como por ejemplo un red de sensores distribuidos o nómadas.

Las redes de sensores están cada vez más extendidas y llegará el momento en que todo objeto tenga conexión con Internet, lo que resulta un riesgo para la privacidad de los usuarios. Por ello a continuación vamos a hablar sobre la seguridad necesaria para que este tipo de servicios puedan percibir la confianza del usuario.

Actualmente, las redes que están en funcionamiento desde hace tiempo son sistemas complejos en los que operar, integrar y seguir avanzando hacia estructuras mejores en todos los aspectos. Existen diferentes tipos de redes, desde móviles hasta satelitáles y fijas, todas unidas por diferentes nodos.

La corriente actual de los operadores son las redes todo IP como núcleo, distribuidas y cada vez más eficientes y rápidas. Entre ellas también es tendencia aplicar la teoría de juegos para colaborar y generar mayor valor que cuándo las teorías se aplican a la competición.

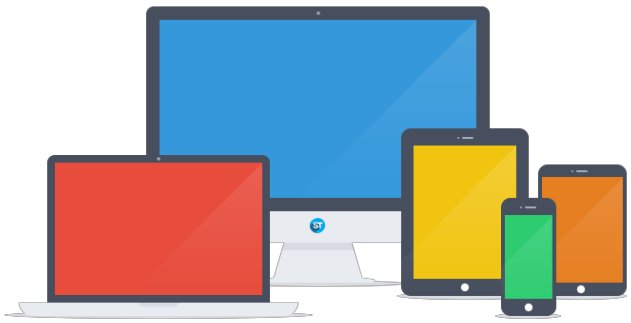
Para el IoT y las redes de sensores todo con dispositivos embebidos tenemos diferentes protocolos. El principal protocolo es CoAP (Constrained Application Protocol) que es muy diferente de HTTP o SIP ya que está diseñado para dispositivos embebidos, pero en esencia ofrece una arquitectura RESTful con la que podremos interactuar en forma de APIs REST fácilmente y sin

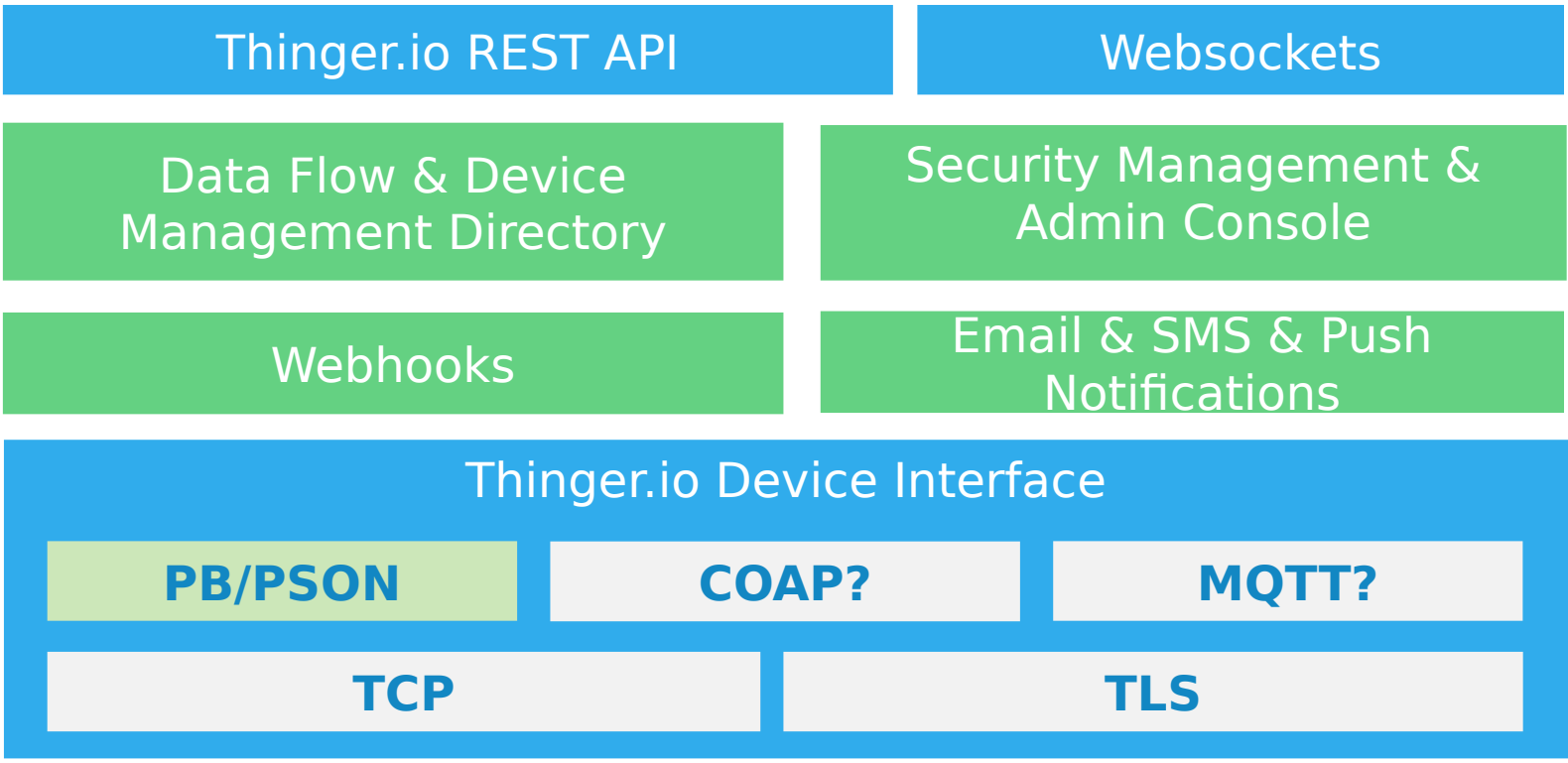
HTTP.

MQTT (MQ Telemetry Transport) es otro protocolo para el IoT. Pero tenemos como alternativa utilizar XMPP (Extensible Messaging and Presence Protocol) que además nos ofrece servicios multimedia al estilo de SIP. XMPP es otro protocolo usado el Io así como en comunicaciones multimedia.

Para direccionar los recursos disponemos de URLs y de URIs, dónde las URIs serán perfectas para el IoT mientras las URL seguirán representando recursos web.

Ahora vamos a ver un ejemplo usado junto a nuestra plataforma que la convierte en una plataforma paralela al Internet de la Cosas, Thinger. Thinger utiliza protocolos propios y arquitecturas propias optimizadas paso a paso. En la siguiente imagen vemos el ejemplo:





En el apartado 3 de las arquitecturas veremos este ejemplo más detalladamente.

2.4.2- Seguridad Informática

En el IoT se tratan datos muy sensibles sobre la privacidad del usuario como su identidades, sus datos personales o profesionales, problema que no cumple con la siguiente definición de un sistema de información de base seguro.

Una definición formal de seguridad en las tecnologías de la información y las comunicaciones es:

> "La seguridad es la capacidad de las redes o de los sistemas de información para resistir, proporcionando un nivel suficiente de confianza, los accidentes o acciones maliciosas que puedan comprometer la disponibilidad, autenticidad, integridad y confidencialidad de los datos almacenados y de los servicios que se ofrecen."

En un análisis de seguridad se ha de seguir un proceso de gestión de riesgos que cumpla con las diferentes dimensiones de la seguridad:

-Disponibilidad. Que los servicios puedan ser usados cuando se requiera de ellos.
-Integridad. La completitud de los datos gestionados por el sistema.
-Confidencialidad. La información debe ser accedida sólo por las personas autorizadas.
-Autenticidad. Que no exista duda de quién se hace responsable de una información o servicio.

En el proceso de gestión de riesgos siempre existirá un riesgo intrínseco y otro residual. El intrínseco es el que se daría si no se tuvieran en cuenta medidas de seguridad. El residual es el riesgo que se da incluso con las medidas de seguridad.

Como ejemplo, primero se haría un análisis de la seguridad del sistema sin medidas de seguridad, obteniendo como resultado el riesgo intrínseco al sistema. Después se intentaría fortificar el sistema con distintas medidas de seguridad y volviendo a hacer la misma auditoría se hallaría el riesgo residual. Por lo que, la diferencia entre ambos riesgos será el riesgo ganado en el proceso de fortificación o de aplicación de las medidas de seguridad.

2.4.2.1-Seguridad Ofensiva

En este capitulo hablaremos sobre los principales ataques que pueden ocurrir en entornos de internet.

2.4.2.1.1-Metodología para Tests de Intrusión

Podemos definir test de intrusión como las pruebas de ataques a un sistema, intentando hacer lo que un atacante verdadero haría. Formalmente, lo llamamos auditoría técnica ofensiva. Existe una clasificación genérica de diferentes tipos de auditoría:

-Caja blanca: En este caso la auditoría se realiza desde dentro del sistema, y por lo tanto se dispone de algunos privilegios de acceso a los sistemas. Se intenta verificar hasta donde se puede llegar con esos ciertos privilegios así como validar el estado y configuración de los sistemas.

-Caja negra: En este tipo de auditoría se tiene la visión del sistema que tendría cualquier persona externa. Por lo que los ataques se realizan desde fuera y únicamente se tiene la información pública del sistema. Por lo tanto, se trata de evaluar la seguridad del sistema perimetral y de los servicios expuestos al público.

-Caja gris: Este caso es un híbrido entre los dos anteriores, a medida del caso en concreto. Realmente es el caso más común ya que no todo es solamente blanco o negro.

En todos los casos estamos hablando de análisis formales definidos por un contrato previo a la ejecución de la auditoría, por lo que el alcance y amplitud del proceso estaría firmado además de definido previamente.

Más específicamente,tenemos metodologías como OSSTMM (Open Source Security Test Methodology Manual), dónde los procesos de intrusión o auditoría tienen una metodología estándar y en el caso de la seguridad web tenemos la metodología OWASP.

Así pues, en general, un test de intrusión se puede dividir en las siguientes fases:

A- Recopilación de Información

En esta fase se trata de buscar información acerca del sistema objetivo. Esta fase está mas orientada a auditorías de caja negra, en las que se busca entre la información pública.
Se pueden utilizar redes sociales, ingeniería social y en general cualquier método que permita un reconocimiento previo del sistema a auditar.

B- Enumeración de Información

En esta fase se trata de obtener datos técnicos concretos sobre el objetivo y listarlos.
Normalmente se utilizan escáneres de servicios para enumerar los servicios disponibles en la red.

C- Análisis: Búsqueda de Vulnerabilidades

En esta fase se buscan posibles vulnerabilidades en el sistema. Para ello ya deberemos saber los servicios que están disponibles en la red, los cuáles ya hemos determinado en la anterior fase.

Una vez sepamos con seguridad los servicios y versiones de los mismos que están en la red, podremos buscar en bases de datos si éstos tienen vulnerabilidades aún no corregidas que más adelante se puedan explotar.

D- Explotación y Post-Explotación

En la fase de explotación y post-explotación se tratara de introducirse en el sistema así como de dejar vías abiertas para tener un acceso persistente.

Para ello, independientemente de las vulnerabilidades que hayamos encontrado en la paso anterior, tendríamos que intentar descubrir contraseñas por fuerza bruta si no hemos conseguido ninguna información sensible anteriormente, o bien buscar o desarrollar programas (exploits) que traten de aprovechar las vulnerabilidades.

E-Documentación

En esta fase se generará un informe con los resultados obtenidos. Se puede distinguir entre informe técnico e informe ejecutivo. Es una fase paralela a todas las demás. En la misma, también se deberán dar las recomendaciones para solucionar los problemas de seguridad encontrados.

2.4.2.1.2-Ataques en Nivel de Red

Principalmente, a nivel de red, tenemos los ataques denominados hombre en medio o MITM (Man In The Middle). Pero también podemos tener ataques por envenenamiento de ARP (Address Resolution Protocol) o ARP Spoofing, de DNS (Domain Name Service) Spoofing, de DHCP (Dynamic Host Configuration Protocol) Rogue, AP (Access Point) Rogue, ICMP (Internet Control Message Protocol) redirect y/o ataques a IPv6 como Neighbour Spoofing.
También tenemos la opción de husmear (sniffar) el tráfico de la red.

2.4.2.1.3-Ataques en Nivel de Servicios

Cualquier servidor ofrece uno o varios servicios a sus clientes.
En este nivel, trataremos ataques contra servicios como HTTP, SSL,TFTP o MySQL o PosgreSQL.

También existen ataques de denegación de servicio (DoS). Estos ataques se pueden realizar con herramientas creadas para probar la carga de los servicios porque también pueden ser utilizadas para saturar el sistema. Y la versión distribuida, DDoS, la más utilizada.

2.4.2.1.4-Ataques en Nivel de Servidor

Los principales ataques a nivel de servidor tratan de explotar vulnerabilidades del sistema operativo que se ejecuta en la maquina servidor. La principal vía es cuando un sistema no está actualizado o configurado correctamente. Aunque lógicamente existen métodos más sofisticados como los que tratan directamente con el núcleo del sistema operativo.

2.4.2.1.5-Ataques en Nivel de Aplicación

A este nivel tenemos múltiples posibilidades. Cada aplicación que se ejecute en el sistema tendrá diferentes vulnerabilidades.
En nuestro caso la aplicación a explotar va a ser Web y en Internet, y en ella tendremos diferentes vectores de ataque como:

-Fraudes:
Los fraudes a través de un sistema se pueden dar una vez tenemos cierto control/privilegios sobre el sistema.
Por ejemplo, podemos realizar una suplantación de identidad en las llamadas telefónicas, llamada Vishing, siendo el phising el caso normal.
También podríamos lanzar Spam telefónico(SPIT) que son llamadas no deseadas, generalmente automáticas, al igual que emails.

-Basándonos en las capturas de la red hechas, podremos utilizar herramientas para extraer los datos de autenticación de los usuarios. Después, mediante diccionario o fuerza bruta se podría extraer información relevante como credenciales de acceso.

-Interceptación del flujo:
La escucha de las comunicaciones entre los extremos implicados.

-Para buscar nuevas vulnerabilidades en sistemas web, al igual que con todos los servicios, existen programas llamados fuzzers, que están diseñados para buscar vulnerabilidades en el sistema. Básicamente tratan de enviar datos mal formados o inesperados por el sistema y así ver como reacciona el sistema ante estos datos arbitrarios. De esta manera se encuentran fallas o errores en los programas objetivo.

-También tenemos posibilidades de hacer secuestro de sesiones o incluso de crear el anonimato.

2.4.2.2-Seguridad Defensiva

En este apartado vamos a hablar sobre la seguridad desde el punto de vista defensivo. El objetivo es, después de conocer los posibles ataques, estudiar cómo proporcionar seguridad a nuestro sistema para evitar todos los ataques. Por lo tanto, se intentarán tomar las medidas correctoras oportunas que se debían tener en cuenta para fortificar el sistema bajo test. Son las

llamadas contra medidas.

La seguridad defensiva se proporciona mediante un proceso de fortificación basado en una teoría o modelo de defensa con tres principios técnicos:

-Defensa en profundidad
-Mínimo privilegio posible
-Mínimo punto de exposición

2.4.2.2.1-Defensa en Profundidad

El modelo de defensa en profundidad proviene del entorno militar, y se refiere a tener múltiples líneas de defensa en vez de una sola muy reforzada. En términos informáticos, este modelo propone crear varias capas de defensa con las que aumentar la probabilidad de detección de ataque y aporta un mayor tiempo para ejecutar la gestión de la defensa.

No hablaremos sobre procedimientos, concienciación, políticas ni seguridad física; que sí serían obligatorios en un plan completo de seguridad TI.
Por todo ello la primera capa de seguridad que vamos a tratar es la de la red interna.

A-Red Interna

Una red interna puede llegar a ser muy compleja teniendo involucrados múltiples sistemas.

Una medida sería la segmentación de la red en función de los usuarios y servicios que se requieran. La red se podría segmentar bien físicamente mediante conmutadores y enrutadores o bien mediante VLANs (Virtual LAN). Las VLANs refuerzan la capa de enlace, ofreciendo protección a ataques MITM y a interceptación de las comunicaciones.

Otra medida muy importante a nivel de red interna es la de disponer de sistemas de detección y/o prevención de ataques, los cuales son llamados IDS (Intrusion Detection System) e IPS (Intrusion Prevention System). En el caso de poner un sistema de este tipo monitorizando la red lo llamamos NIDS (Network IDS). Estos sistemas pueden ser, en función de cuando actúan, reactivos o preventivos, donde los reactivos (IDS) detectan intrusiones ya ocurridas y los preventivos (IPS) capturan la intrusión en el momento y toman medidas para bloquearla. Además, también se pueden dividir en función de en base a qué datos toman las decisiones; se pueden basar en firmas de red , en comportamientos o en firmas en los logs.
A priori, parecen elementos muy útiles pero también tienen ciertos inconvenientes, como por ejemplo, los posibles falsos positivos.

Un elemento asociado a estos, sería un cortafuegos para la red, que corresponde a la zona perimetral, la cuál suele estar dividida en capas para ejercitar la defensa en profundidad.

También tenemos medidas de seguridad formadas por la configuración de los servicios de red, como por ejemplo, ARP, DHCP o DNS.

Por último, deberíamos tener a un operador analizando la red con una herramienta de

monitorización. Con esto reduciríamos los falsos positivos e incrementaríamos la eficacia de las respuestas ante ataques. Por ejemplo, podemos utilizar sistemas SIEM (System Information and Event Management).

B- Nivel de Servidor

La seguridad a nivel de servidor puede ser entendida de diferentes maneras. Aquí vamos a tratar de que no llegue a juntarse con la seguridad a nivel de aplicación y servicios.
En esta capa se tendrían en cuenta las actualizaciones del sistema operativo del servidor, con el objetivo de tener el servidor en un estado sólo vulnerable ante ataques basados en zero days (vulnerabilidades no conocidas).

También se ha debe auditar el uso, por parte del administrador, del servidor en local y en remoto para poder llevar un registro completo de lo que ocurre en el servidor, por ejemplo los accesos al mismo.
Otra medida será la inclusión de un HIDS (Host IDS) en el servidor.

C- Nivel de Aplicación

Esta capa es muy importante a la vez que compleja. En ella trataremos sobre las aplicaciones y servicios que se ejecutan en el servidor así como sobre la configuración del cliente, en nuestro caso un navegador web (aunque no sobre el sistema operativo en el que es ejecutado).

Las aplicaciones y/o servicios configurados por defecto son potencialmente una vulnerabilidad, por lo que para corregir esto hemos de configurar las aplicaciones de manera que cumplan las buenas prácticas en cuanto a la seguridad establecidas para cada una, y, por supuesto, manteniendo la funcionalidad del sistema.

Lo más importante será tener las aplicaciones actualizadas para evitar en lo posible ataques zero day y sus posteriores procesos de explotación y elevación de privilegios.

Se ha de tener una política de usuarios basada en privilegios, cumpliendo el principio de mínimo privilegio posible que veremos a continuación.

Además, se ha de llevar una política de máxima restricción para los servicios.

La principal herramienta que se puede explorar para este nivel es la creación de jaulas o sandboxs si lo permite la aplicación.

También deberemos utilizar cortafuegos a nivel de aplicación así como el anti golpeo de puertos (portknocking) y medidas de monitorización como en las anteriores capas.

D- Nivel de Información

La información relevante es la transmitida en la comunicación y la que guarda el computador por lo que son las que deberemos asegurar.

Las técnicas que se utilizan para salvaguardar la información son todas las técnicas contempladas en la criptografía.

2.4.2.2-Mínimo Privilegio Posible

En el modelo de seguridad en el que nos estamos basando, el mínimo privilegio posible sería el segundo principio a tener en cuenta.

Básicamente se trata de que tanto los procesos como los directorios sean ejecutados o accedidos por usuarios con privilegios especificados para ello y siempre guardando las identidades de los mismos.

Formalmente se puede decir que, sólo y solamente si se necesitan elevar privilegios para una acción determinada, estos se eleven únicamente para esa acción concreta en ese momento.

2.4.2.3-Mínimo Punto de Exposición

Se dice que visto el sistema desde el exterior, éste ha de tener expuestos los mínimos recursos posibles, intentando tender hacia el mínimo riesgo posible. Cuanto menor sea el punto de exposición, menor será la amenaza que habría que mitigar o menor será el impacto que se podría sufrir.

Para esto, el sistema ha de tener funcionando únicamente los servicios que se requieran, el software imprescindible para el correcto desempeño de sus funciones.

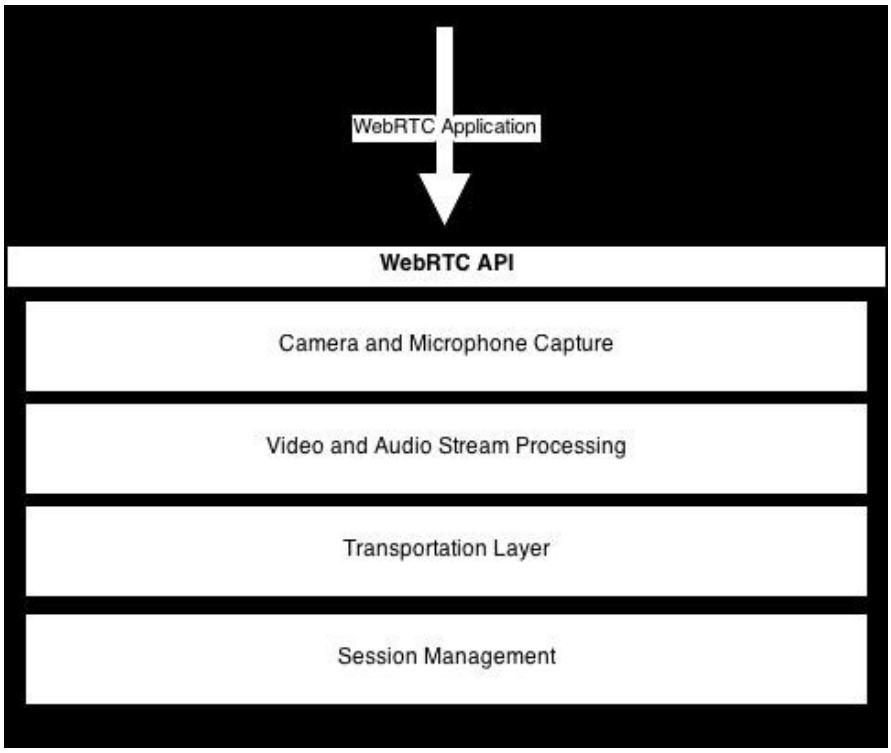
No es recomendable ejecutar varios servicios en la misma máquina, a no ser, que tengan un buen aislamiento entre ellos con, por ejemplo, maquinas virtuales o jaulas. Para hacer una distribución eficaz, habría que ir separando los servicios y clasificándolos en críticos, medios y bajos, en cuanto al riesgo posible en un supuesto ataque.

Otra medida para minimizar el punto de exposición es la segmentación de redes, de cual ya hemos hablado anteriormente.

Para cubrir también la seguridad perimetral, tendríamos que hablar de los cortafuegos ya mencionados y de las zonas desmilitarizadas o DMZ, que no son más que un conjunto de cortafuegos actuando en conjunto.

2.4.3- WebRTC [ref. 7,1,10]

Esta es la estructura de un sistema WebRTC sobre IP:

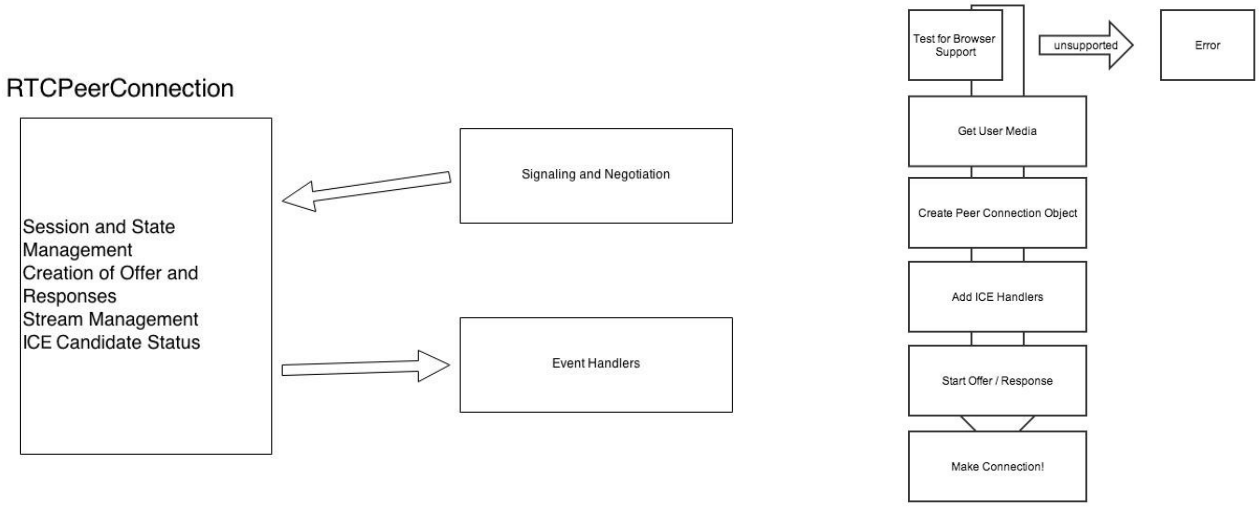


En esta figura extraída de la bibliografía, vemos como podemos usar XMPP o SIP para manejo de sesión, TCP o UDP para transporte, un ProceSiX que procese el Stream o flujo, una cámara y un micrófono y con todo esto podremos interactuar con las APIs estándar de WebRTC y complementarias como las que usará ProceSiX que serán de audio y vídeo.

No se nos olvide un navegador web compatible con los últimos estándares del W3C y del IETF o el IEEE para que todo pueda funcionar correctamente.

Después estar al día de las innovaciones en WebRTC desde su salida hasta el día de hoy, tenemos una visión global y de última tendencia. Por esto, a continuación de los siguientes esquemas introducimos los resúmenes de los artículos del estado del arte o drafts sobre todo lo relativo a WebRTC o RTCWEB según la entidad estandarizadora.

Para hacer una conexión peer to peer, aquí tenemos el diagrama de flujos de la comunicación de uno de los peers:



Para comunicaciones como salas de reuniones virtuales tenemos los MCUs que operan con SIP y con P2P.

Pero, a continuación vamos a ver los resúmenes más importantes de los últimos drafts:

-***Overview:*** *Real Time Protocols for Browser-based Applications draft-ietf-rtcweb-overview-14:*

"This document gives an overview and context of a protocol suite intended for use with real-time applications that can be deployed in browsers - "real time communication on the Web". It intends to serve as a starting and coordination point to make sure all the parts that are needed to achieve this goal are findable, and that the parts that belong in the Internet protocol suite are fully specified and on the right publication track. This document is an Applicability Statement - it does not itself specify any protocol, but specifies which other specifications WebRTC compliant implementations are supposed to follow.

This document is a work item of the RTCWEB working group."

- ***Security Considerations*** *for WebRTC draft-ietf-rtcweb-security-08:*

"The Real-Time Communications on the Web (RTCWEB) working group is tasked with standardizing protocols for real-time communications between Web browsers, generally called "WebRTC". The major use cases for WebRTC technology are real-time audio and/or video calls, Web conferencing, and direct data transfer. Unlike most conventional real-time systems (e.g., SIP-

based soft phones) WebRTC communications are directly controlled by a Web server, which poses new security challenges. For instance, a Web browser might expose a JavaScript API which allows a server to place a video call. Unrestricted access to such an API would allow any site which a user visited to "bug" a user's computer, capturing any activity which passed in front of their camera.

This document defines the WebRTC threat model and analyzes the security threats of WebRTC in that model."

- ***Using ZRTP*** *to Secure WebRTC draft-johnston-rtcweb-zrtp-02:*

"WebRTC, Web Real-Time Communications, is a set of protocols and APIs used to enable web developers to add real-time communications into their web pages and applications with a few lines of JavaScript. WebRTC media flows are encrypted and authenticated by SRTP, the Secure Real-time Transport Protocol while the key agreement is provided by DTLS-SRTP, Datagram Transport Layer Security for Secure Real-time Transport Protocol. However, without some third party identity service or certificate authority, WebRTC media flows have no protection against a man-in-the-middle (MitM) attack. ZRTP, Media Path Key Agreement for Unicast Secure RTP, RFC 6189, does provide protection against MitM attackers using key continuity augmented with a Short Authentication String (SAS). This specification describes how ZRTP can be used over the WebRTC data channel to provide MitM protection for WebRTC media flows keyed using DTLS-SRTP. This provides users protection against MitM attackers without requiring browsers to support ZRTP or users to download a plugin or extension to implement ZRTP."

- ***Transports*** *for WebRTC draft-ietf-rtcweb-transports-09:*

"This document describes the data transport protocols used by WebRTC, including the protocols used for interaction with intermediate boxes such as firewalls, relays and NAT boxes."

- ***Web Real-Time Communication*** *(WebRTC): Media Transport and Use of RTP draft-ietf-rtcweb-rtp-usage-25:*

"The Web Real-Time Communication (WebRTC) framework provides support for direct interactive rich communication using audio, video, text, collaboration, games, etc. between two peers' web-browsers. This memo describes the media transport aspects of the WebRTC framework. It specifies how the Real-time Transport Protocol (RTP) is used in the WebRTC context, and gives requirements for which RTP features, profiles, and extensions need to be supported."

- *WebRTC* ***Gateways*** *draft-ietf-rtcweb-gateways-01:*

"This document describes interoperability considerations for a class of WebRTC-compatible endpoints called "WebRTC gateways", which interconnect between WebRTC endpoints and devices that are not WebRTC endpoints."

- *WebRTC* ***Video Processing and Codec Requirements*** *draft-ietf-rtcweb-video-06:*

"This specification provides the requirements and considerations for WebRTC applications to send

and receive video across a network. It specifies the video processing that is required, as well as video codecs and their parameters."

- *WebRTC* ***Audio Codec and Processing Requirements*** *draft-ietf-rtcweb-audio-08:*

"This document outlines the audio codec and processing requirements for WebRTC endpoints."

2.6- MachineToMachine (M2M): Raspberry Pi y Arduino

[ref. 7.1.6] [ref. 7.1.7] [ref. 7.1.8] [ref. 7.1.9] [ref. 7.1.10]

2.6.1- Arduino

2.6.1.1- Hardware

Físicamente Arduino es un microcontrolador que está rodeado por diferentes bloques que dependen de la versión de la placa que a su vez puede estar rodeada por más placas iguales o de diferentes versión. Habitualmente están compuestas por el sistema de alimentación, los pines de alimentación, los pines analógicos, los digitales, LEDs, el reset, el puerto ICSP, el sistema USB y por supuesto el microcontrolador.

Los periféricos internos del microcontrolador son el sistema de reloj, temporizadores o contadores, el sistema PWM (Pulse Width Modulation), el bus I2C, el interfaz SPI, el puerto USART, el ICSP, el USB y los conectores de los pines. Posteriormente los veremos con más detalle.

Arduino se está convirtiendo en un estándar de facto debido al extenso uso que se está dando de él en todo el globo terráqueo. Es hardware abierto que permite diseñar placas personales que sean compatibles con todas las demás compatibles y propias de Arduino. Su principal característica es el bajo coste y bajo consumo siendo la tendencia que se conviertan en sistemas ARM de 32 bits capaces de computar información a gran velocidad y bajos costes. Cualquier placa compatible con Arduino puede ser añadida al microcontrolador con facilidad, llamadas escudos o shields. Y debido a su gran versatilidad Arduino puede ser integrado con otros sistemas de cómputo o electrónicos fácilmente.

2.6.1.2- Software

Arduino, a parte de ser hardware abierto, también es un entorno completo de programación desde el IDE hasta el compilador y el mismo lenguaje. Este lenguaje tiene gran versatilidad ya que puede incorporar integraciones con otros lenguajes en el mismo IDE. Principalmente, el código fuente está escrito en C/C++ y Java, aunque tiene compatibilidad con Processing, Wiring, Python y Javascript mediante diferentes cadenas de traducción de código a los

distintos lenguajes.

El IDE es portable ya que está basado en Java, y con él se escribe el software y se verifica, compila y carga directamente al la placa hardware correspondiente mediante diferentes métodos de programación de microcontroladores.

También existen otros entornos de programación que permiten programar una placa Arduino mediante plugins a entornos específicos como Visual Studio, a IDEs como Minibloq, Pduino, Marimole o Codeblocks. Pero también han aparecido simuladores para Arduino como Simuino, Ardusim, Emulino, VBBExpress, Emulare y Vitronics.

2.6.1.3- Comunicaciones

Como hemos visto mediante los periféricos internos del microcontrolador podremos comunicarnos con el exterior mediante los buses de comunicaciones que veremos a continuación.

USART (Universal Synchronous and Asyncronous Serial Receiver and Transmitter) es un dispositivo para realizar comunicaciones serie. Utiliza el formato TTL muy parecido al RS-232. Se utiliza un pin para transmisión y otro para recepción.

Otros tipos de comunicaciones serie populares son I2C (Inter Integrated Circuit) y SPI (Serial Peripheral Interface). Éstos se pueden usar como buses compartidos entre varios dispositivos que serían los esclavos del microcontrolador como maestro. I2C usa un canal para los datos y otro para el reloj. SPI está formado por el canal de reloj, el de Chip Select, la salida MOSI y la entrada MISO.

En Arduino existe la clase Stream de la que heredan la clases Serial, Wire, Ethernet y SD, con las que manejar los flujos de información.

También se puede usar buses industriales, como CAN, que junto a su placa convertidora ya que la mayoría de Arduinos no los implementan, nos permite adaptarnos a aplicaciones industriales.

2.6.1.4- Periféricos

El microcontrolador es como un microprocesador de baja potencia e incorpora en su misma placa otros tipos de periféricos nombrados anteriormente.

El propio microcontrolador tiene una fuente de reloj utilizada para la temporización de todo el sistema, desde la frecuencia de ejecución de instrucciones hasta contadores o timers.

Las interrupciones pueden ser internas o externas y ofrecen un

comportamiento basado en detección de eventos. El principal contador es el watchdog que además de poder lanzar interrupciones puede resetear la CPU, por ejemplo en situaciones de bucle infinito.

Estamos tratando con dispositivos que consumen poco debido a que suelen estar embarcados en sistemas portátiles dónde interesa hacer que alimentación sea lo mas eficiente posible. Para ello debemos de dormir los procesos de los periféricos que no se usen como la CPU cuando está ociosa. Mediante un dispositivo como un contador encendido se lanzaría una señal que despierta el bloque a usar a continuación.

La mayoría de microcontroladores siguen la arquitectura Hardvard, en la que en el mismo chip se integra una memoria para el programa y otra para los datos. Podemos diferenciar entonces la memoria RAM (volátil) de la Flash (permanente) y de la EEPROM (persistente).

2.6.1.5- Librerías

Para mejorar en el software en términos de rapidez, facilidad de implementación y portabilidad utilizaremos librerías. Con ellas podremos programar a más alto nivel el sistema. Gracias a la comunidad open source están disponibles una inmensidad de librerías con las que abstraer los detalles de bajo nivel para crear sistemas más complejos mediante programación orientada a objetos. Además, para cada hardware compatible con Arduino hay varias librerías que implementa las funciones utilizadas para las aplicaciones correspondientes.

Por ejemplo, podemos tener librerías para memorias SD, puertos Ethernet, RFID, Radio AM y FM, antenas móviles GSM/UMTS, WiFi, ZigBee, Pantallas TFT, Motores y sensores de cualquier tipo.

2.6.1.6- Integración con otras plataformas

Una de las ventajas mas interesantes es que Arduino se puede integrar casi con cualquier plataforma con diferentes métodos de comunicación.

El StandardFirmata es la técnica más utilizada ya que con Firmata se puede integrar Arduino con ZigBee, Bluetooth, Android y otros lenguajes como Python, .NET, LabVIEW, MatLab, C++ y con servicios web. Los servicios web pueden ser SOAP o REST, teniendo datos en formato JSON o XML. Con RESTduino se puede tratar con este tipo de comunicación.

Internet de las cosas es un paradigma integra diferentes soluciones hardware y software de manera transparente al desarrollador. Se hace una integración con Arduino mediante librería por ejemplo en C11 o C99, mediante servicios web como REST o para flujos multimedia con websockets. La conexión entre la plataforma de servicios distribuida en Internet y los dispositivos que generan y consumen datos es mediante TCP o UDP a un puerto del dispositivo en cuestión. El paso de NATs y Firewall es transparente por diseño pero un elemento como un router puede actuar como máster al igual que los dispositivos y actuar como servidor de servicios para cosas más pequeñas o simplemente dentro de la red interna y no expuesta a Internet para circuitos

cerrados. Para identificar cada dispositivo asociado al servicio se utilizan metadatos que describen el dispositivo así como el tipo de acceso a la plataforma de IoT.

2.6.1.7- Núcleo Arduino

A- ToolChain

La cadena de desarrollo de Arduino está formada por el preprocesador, el compilador y el enlazador.

El preprocesador transforma el código Arduino a C++ con el preprocesador de Processing. Como resultado obtenemos código C++, lo que permite utilizar todas las herramientas de éste lenguaje.

El compilador es la variante de gcc para microcontroladores AVR. Así para la configuración del mismo debemos editar los ficheros makefile. Se compilarán por un lado los Sketchs y por otro los ficheros de C y de C++, además de tener ya incluidas todas las rutas necesarias. Como resultado de este proceso obtenemos varios ficheros objeto “.o” en el directorio temporal del proyecto.

Con el enlazador o linker se unirán todos los ficheros objetos y se posicionarán según las direcciones de memoria de cada uno. Como resultado se obtiene el código máquina que ejecutará el microcontrolador en formato Intel HEX.

Una vez tenemos los Sketchs en formato HEX será el Bootloader quién se encargará de grabar en la memoria flash del microcontrolador el programa ejecutable.

La gran ventaja del entorno Arduino es que podemos configurar a nuestro antojo todo el proceso software adaptándolo a nuestras necesidades. Así, podemos crear nuestras propias librerías o incluso nuestros propios circuitos integrados o placas.

2.6.2- Raspberry Pi

2.6.2.1- Hardware

Raspberry Pi es una fundación que produce diferentes modelos de hardware abierto para educación en informática para cualquier nivel de conocimientos incluso para investigación en ciencias de la computación. El juego de instrucciones utilizado en todos los modelos es RISC de 32 bits.

La versión de hardware más moderna es la 2, que tiene 4 núcleos de

procesador ARMv7 a 700MHz cada uno, junto a 1 GB de RAM, una GPU integrada y diferentes interfaces o periféricos. Estos son 4 puertos USB, 1 Ethernet, 1 HDMI, 1 Jack de 4 pines, una ranura para microSD, un conector para cámara MIPI CSI, un conector DSI para pantallas LCD y un puerto GPIO.

La versión 1 B+ tiene los mismos interfaces que su superior pero únicamente tiene 512 MB de memoria RAM y 1 procesador ARM1176JZF-S, (ARM11).

Ambos modelos son SBC (Single Board Computer) Broadcom BCM2836 y BCM2835 respectivamente. Este sistema integrado incorpora como subsistemas la CPU, la GPU, el DSP, memoria SDRAM compartida y el hub USB con Ethernet incluido. El consumo energético varia entre 2 W y 4 W siendo 5 V la alimentación. El puerto GPIO incluye SPI, I2C y UART, vistos con Arduino.

2.6.2.2- Software

Hay muchos sistemas operativos que soportan procesadores con arquitectura ARM, pero para Raspberry Pi sobresalen Raspbian, Pidora, Arch Linux, IPFire, Kali Linux, Micro Elastix, familia BSD, RISC OS y FreeRTOS entre otros. La versión 2 es también soportada por más sistemas operativos como Ubuntu Snappy Core, Windows 10 o Ubuntu Mate. Actualmente están en camino de ser soportados muchos otros sistemas operativos como FirefoxOS o Android.

Además, tenemos la posibilidad de configurar el núcleo o kernel del sistema para adaptarlo a nuestras necesidades.

Raspberry promueve los lenguajes open source, principalmente Python, pero soporta C/C++, JavaScript, Java, Perl y Ruby entre otros y cualquier cadena de software que soporte el sistema operativo concreto.

Al ser un computador de bajo coste y bajo consumo así como integrado en una pequeña placa nos aporta muchas ventajas adaptadas a plataformas de IoT.

También nos ofrece lo que cualquier ordenador de sobremesa como son software ofimático, navegadores web, televisores digitales, software de audio, sistemas de ventanas así como cualquier otro software compilado específicamente para Raspberry.

2.6.2.3- Integración

Con todas las comunicaciones y periféricos posibles a bordo de la Raspberry, ésta se puede comunicar incluso con escudos, al igual que Arduino, con cualquier componente informático o electrónico.

Hay también librerías específicas para Raspberry pero como en ella se utilizan lenguajes muy populares como Python se dispone de una ingente cantidad de librerías en

Internet.

Además, estas librerías pueden ser consumidas mediante servicios web como arquitecturas REST sin necesidad de estar instaladas en la propia Raspberry que habitualmente hace cliente de algún o algunos servidores en Internet a pesar de que también es un magnifico servidor casero, que sobre todo es de bajo consumo y bajo coste.

2.6.3- Computación Intensiva

Aquí llamaremos computación intensiva a sistemas que ejecutan procesos de forma distribuida o paralela.

Hemos visto como puede existir un conjunto de sensores que son esclavos de una placa Arduino. Así, también puede haber varios Arduinos que son esclavos respecto a por ejemplo una Raspberry Pi. Y, a su vez, ésta Raspberry Pi puede ser un nodo de computación esclavo respecto a otra Raspberry que actúa como Master.

Por lo tanto, siguiendo esta línea, tendríamos un cluster o grupo de Raspberry Pi para computar información formada por varios nodos de computo y uno de Master. El principal protocolo a tener en cuenta en éste cluster sería MPI (Message Passing Interface) con el que se produce la comunicación entre nodos. Y, como hemos dicho, éste cluster podría actuar de back-end respecto a una red de Arduinos que gestiona una red de sensores.

Ésta arquitectura es la más obvia, pero podemos desplegar arquitecturas híbridas totalmente adaptadas a las necesidades de la aplicación objetivo.

2.7- Tratamiento Digital del Audio en Telecomunicaciones

[Ref. 7.1.2] [Ref. 7.1.5]

En telecomunicaciones el audio es un medio muy importante en las comunicaciones. Desde la llegada del teléfono la voz representa el lenguaje en el que se basan las aplicaciones de audio, un poco mas complejas.

La calidad varía desde codificadores GSM, mp3 o calidad CD de audio profesional. Para enviar flujos multimedia por Internet se utiliza el protocolo RTP que puede utilizar variedad de codecs aunque principalmente se utiliza MPEG-2.

La TV por Internet es un fenómeno retrasado con respecto a otras tecnologías como la misma web. Pero basa su funcionamiento en el envío multimedia de vídeo y audio al espectador quién puede a través de un canal de retorno interactuar con el programa.

La televisión digital también se ha estandarizado por medio inalámbricos de broadcast como tradicionalmente. Ahora todo es digital.

La red es el factor más importante a tener en cuenta en aplicaciones con requisitos temporales multimedia. Tenemos capas que gestionan la sesión como IMS mediante SIP, similar a HTTP. Y MPLS en las troncales con QoS contratada previamente. Todo esto forman redes de nueva generación, NGN.

2.7.1- Fundamentos de Audio Digital

A-Conversión Analógico a Digitales

La cadena de conversión analógico digital es la que captura la realidad física y la transforma a dimensiones finitas.

Está compuesta por: un Filtro Paso bajo Antialiasing, una etapa de Muestreo y Retención a diferentes tasas, y el propio convertidor analógico digital identificado por la etapa de cuantificación se caracteriza por N bits = 2^N amplitudes o niveles. En esta etapa lo más importante es el error introducido por el cuantificador que marca el nivel señal a ruido, SNR. Para solventar el error de cuantificación tenemos diferentes técnicas: aumentar N, Sobremuestreando en A/D, con la técnica del Dither o sobremuestreando en la etapa DAC. El sobremuestreo expande el espectro haciéndole más bajo y más ancho y aumentando así la SNR. La técnica del Dither aumenta la cantidad de ruido a cambio de tapar el ruido de tipo distorsión y se suma antes del muestreo. Esto es menos molesto ya que el ruido blanco a bajo nivel es relajante.

Para todo este proceso necesitamos un generador de ruido pseudo-aleatorio digital que también podremos usar como hardware en otras partes del sistema general del prototipo. Por ejemplo, podemos generar semillas de claves simétricas o asimétricas para criptografía y por ejemplo cifrar todos nuestras comunicaciones con semilla propia.

B- Conversión Digital a Analógico

La cadena de conversión digital analógico es la que transforma de dimensiones finitas a infinitas de vuelta.

Está compuesta por: el propio convertidor, una etapa de muestreo y retención con un deglitcher, un filtro paso bajo recuperador y un filtro recuperador digital. El principal problema en esta etapa D/A son glitches o transisciones en la señal pero para solucionarlo tenemos el deglitcher

comentado. También necesitaremos un generador de ruido plano con buenas características. Los filtros analógicos se complementarán con filtros digitales FIR o IIR.

C- Aritméticas de Punto Fijo y Flotante para DSP

Se ha de encontrar la longitud de palabra óptima tanto en el subsistema A/D como en almacenamiento y reproducción así como en procesamiento interno de la señal en el DSP. En este DSP se pueden producir errores por desbordamiento, de redondeo y en los coeficientes.

La representación digital de las muestras pueden ser en coma flotante o en coma fija. Las de coma fija pueden ser 8 bits con o sin signo pero también podemos aumentar los bits a 16 y 32 o 24 y 48. La coma flotante se basa en la notación científica representada por una mantisa y una base. La coma flotante aporta mayor rango de representación de números por lo que los DSPs que trabajan en coma flotante suelen ser más caros y complejos.

Por lo tanto, según las aplicaciones los sistemas de coma flotante son buenos dónde los coeficientes varían con el tiempo (Sistemas adaptativos) y dónde las señales y coeficientes tienen un gran rango dinámico. Si la aplicación no tiene gran rango dinámico será menos costoso implementar sistemas de coma fija.

D- Implementación de Filtros y Ecualizadores

Basados en la teoría de sistemas lineales podemos implementar filtros y ecualizadores digitales. Los filtros pueden ser IIR o FIR. En ambos los parámetros de la función son: K frecuencia de corte o sintonía, factor de calidad Q y realce o atenuación V0.

Los ecualizadores pueden ser resonantes (peak filters) y de alta y baja frecuencia (shelving filters). Además pueden ser de 1º , 2º o 3º orden. También podemos conectar diferentes subsistemas en cascada o serie o en paralelo dando como resultado bancos de filtros también llamados ecualizadores. En el siguiente apartado en el procesado de frecuencia veremos con más detalle los ecualizadores. Únicamente remarcar que en todos podremos variar los tres parámetros expuestos al principio.

2.7.2- Procesado Digital de Audio

A- En frecuencia

En frecuencia podemos transformar la señal y sus respuesta en este dominio a nuestro antojo. Tenemos ecualizadores fijos, correctores de tonalidad y ecualizadores sintonizados.

Los fijos pueden ser paso bajo (Rumble o Subsónico), paso alto (Scratch), MPX (suprime 19KHz), los correctores de gananacia y los crossover o filtros de cruce pasivos (utilizados en altavoces de varias vías).

Los correctores de tonalidad ofrecen una modificación de la ganancia/atenuación en términos de control de graves y agudos, control de medios, control de presencia y control de sonoridad fisiológica (Loudness). Ofrecen un reforzamiento variable para salas multimedia.

Los Ecualizadores sintonizados pueden ser Gráficos (combinación filtros paso banda de sintonía), Paramétricos (controlan todos los parámetros del ecualizador), Semiparamétricos (controlan ganancia y sintonía) o Bancos de Filtros de Ranura (atenuación ruido y señales parásitas).

B- De Dinámica

Los procesadores de dinámica son equipos destinados a alterar los rangos dinámicos de la señal de audio consiguiendo adaptarla a condiciones específicas o para producir efectos sonoros.

Entre ellos tenemos el Compresor (lineal, de ganancia constante o limitador), el Expansor (lineal o de ganancia constante), la Puerta de Ruido (Expansor especial), De-Esser (señales de voz) y el Compander (Compresor + Expansor).

C- Temporal

En el procesado temporal tenemos Líneas de Retardo Electrónicas para generar Reverberación Artificial y otro Efectos como Chrorus, Doubling, Flanging, Vibrato, Efectos Multibanda o combinaciones de todos los anteriores.

La reverberación artificial sería la más importante pues puede controlar si el campo que el oyente percibe es más directo que difuso. Así simulamos entornos acústicos a medida determinando la impresión espacial del sonido . Lo ideal es implementar una reverberación con respuesta plana.

3- Diseño de las Arquitecturas

La implementación del sistema y de los subsistemas correspondientes es el principal gran avance en el contexto de este proyecto.

Cada subsistema tiene sus propias peculiaridades, clasificadas según los requisitos funcionales de cada cuál. A continuación, vamos a detallar los problemas encontrados al implementar las plataformas con su arquitectura correspondiente.

3.1- Requisitos de la Arquitectura

Estamos tratando con una arquitectura multimedia que impone requisitos de tiempo, de prioridad, de ancho de banda y resto de factores importantes en aplicaciones semejantes. La tendencia es hablar de tiempo real ya sea en el kernel o en Internet, pero las aplicaciones multimedia no necesitan tiempo real sino bajo retardo o latencia y bajo jitter, así como lo comentado anteriormente.

También es una arquitectura orientada al dato o al recurso, mediante JSON, APIs REST, CoAP, Protoson (Thinger) y bases de datos NoSQL como Cassandra, MongoDB, CouchDB. También habrá bases de datos relacionales con MySQL y PostgreSQL. Para interactuar con la base de datos desde el cliente hará falta una comunicación bidireccional en JSON y en WebSockets que hablará con la parte del núcleo de ProceSiX que estará implementada con un ORM con Python.

El cliente estará basado en Angular.js y en Ionic o Cordova para móviles. También tenemos clientes basados en LAMP con PHP, Javascript, HTML5 y CSS3, así como estáticos con librerías Javascript como jQuery o ya volvemos a Angularjs e Ionic Framework.

El middleware podrá ser Express.js del stack MEAN, o del stack Python tenemos muchos elegidos por la tremenda disponibilidad de librerías en el entorno Python. Ya las tenemos elegidas.

Los servidores pueden ser Node.js de MEAN o Tornado, Gevent, Gunicorn, Twisted para el caso de la pila de Full Stack Python. El servidor de Odoo, CMS, no es ninguno de éstos pero también es uno conocido en el mundo de bibliotecas Python.

Django será el framework full stack Python y MVC e integrable con la web como con Angularjs e Ionic así como con Odoo ERP, Thinger y páginas estáticas y servicios especiales de empresas conocidas como Google , Facebook, Apple, Microsoft y el más importante en nuestro caso sería la infraestructura como servicio, IaaS, que ofrece Amazon.

La integración en Django surge mediante el modelo de Aplicaciones o APPs dónde tenemos un repositorio con todas las disponibles. Ya tenemos elegidas la mayoría de librerías pero en cualquier caso remito a la bibliografía al libro “Two Scoops of Django 1.8”, que ofrece una completa visión de las mejores prácticas en el mundo de las APPs y librerías de Django. La más importante es Django_Rest_Framework que será parte de nuestro core así como librerías de entrada y salida asíncronas como Tornado o Twisted, aunque en este sentido nuestra opción será integrar un sistema asíncrono dentro de Django ya que éste lo permite.

Con Django como framework MVC implememtaremos el sistema. El Modelo será ProceSiX, el Controlador Thinger y las Vistas serán variadas principalmente con tecnología Angujarjs e Ionic aunque también páginas estáticas.

3.2- Planteamiento del Diseño

En el planteamiento del diseño hemos tenido en cuenta muchos factores para tomar las decisiones sobre qué y cómo hacer las actividades desarrolladas. Lo principal es el núcleo o core del que veremos que está formado por una arquitectura en panal hexagonal que incluye bases de datos y sistemas operativos, servicios web y otras abstracciones para interactuar con terceras partes.

El planteamiento del sistema ha sido totalmente autónomo pero a raíz de conocer Thinger.io ofreciendo la posibilidad de colaborar nos ha sido mucho más sencillo realizar el proyecto. Las semejanzas entre nuestro diseño de ProceSiX con el de Thinger son muchas ya que ambas siguen los estándares y últimas tendencias de Internet. Pero la gran diferencia es la función que ejerce cada uno en una aplicación compartida. Thinger monitoriza, controla y comanda el Internet de las Cosas, mientras que ProceSiX ofrece un sistema para Big Data, Aprendizaje Automático e Inteligencia Artificial Multimedia y Multimodal abstraído en la nube como IaaS en local con el HPC de Raspberrys Pi implementado en Python como intérprete o como PaaS y SaaS con la opción distribuida en Internet de PrceSiX.

En las pruebas de concepto que veremos en el apartado 4 podremos realimentar con pruebas las arquitecturas teóricas presentadas.

La primera es la Interfaz Acústica que impone restricciones de baja latencia para capturar y reproducir audio y está contemplada en la arquitectura Hardware así como en la Audio Work Station.

La Interfaz Lumínica cumple con la arquitectura del Hardware Eléctrico para las bombillas y de Arduino y Raspberry para los pines digitales con PWM para diodos LEDs.

La Interfaz Visual está embebida en la web y es servida por un servidor que forma parte de los repositorios de Debian por lo que encaja perfectamente en nnuestras arquitecturas.

La Interfaz de Control y Monitorización Web está desarrollada cumpliendo lo mostrado en la arquitectura de Software. Tenemos en ésta arquitectura nuestra GUI y también tenemos dentro de Thinger su propia GUI.

El Servicio Reconocimiento de Voz Remoto está delegado a una organización, BioID, que ofrece biometría en la nube con APIs REST.

La Seguridad Web y Odoo ERP, son nuestro contraste de seguridad frente a la seguridad de nuestro HPC local y las raspberrys y arduinos. Cumplimos la arquitectura planteada.

De Computación Intensiva no hay ninguna arquitectura porque es la que se da en el cluster pi o HPC con las 2 raspberrys conectadas por ethernet al mismo router. Lo que si que hay es un anexo dónde se detalla cómo programar las raspberrys para obtener imágenes del sistema operativo configurado que expandirlo hacia más nodos ya sean maestros o esclavos.

La última prueba de concepto pero en la web será para el audio aunque también habrá espacio para el vídeo. Lo primero es la manipulación del audio con el Web Audio API estándar y Javascript a nuestro antojo. La segunda prueba incluida en esta es enviar ese audio manipulado bien a los altavoces de la misma persona o los de otra del otro extremo de la conexión, además junto al audio se enviará el vídeo formando dos canales multimedia WebRTC y un de datos webRTC.

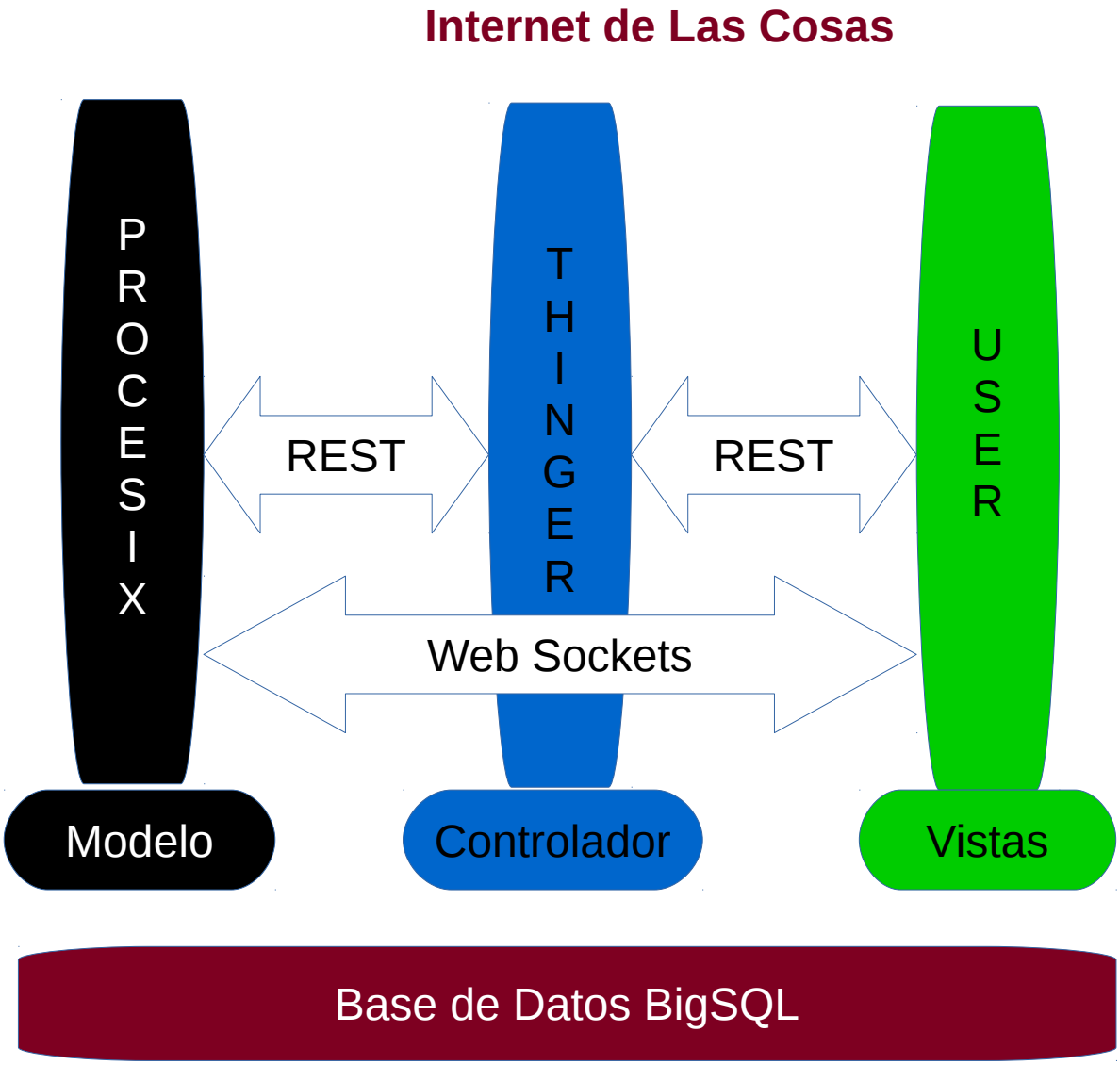
El diseño está planteado con vistas a todas las líneas futuras concluidas en el apartado correspondiente. Por lo tanto, con este elenco de arquitecturas podremos poner en producción todas las líneas futuras propuestas en la memoria.

3.3- Arquitecturas

3.3.1- Arquitectura Principal

Ésta arquitectura tiene como núcleo el paradigma de computación del Internet de las Cosas dónde la información fluye mediante servicios web de un extremo a otro. En el extremo cliente tendremos al usuario y su ambiente, que serán los que generen los datos, lo podemos ver como la base datos de nuestro sistema. Mientras, en el extremo remoto tendremos sistemas que hacen de controlador del sistema cliente, que podrán ser asistentes humanos o asistentes artificiales.

Este esquema está diseñado con la arquitectura MVC (Modelo-Vista-Controlador) para facilitar la integración con todas las posibles aplicaciones que pueden funcionar enmarcadas dentro del mismo y con servicios web como forma de intercambio de información estructurada.



Tenemos a la derecha al usuario que interactúa con la vista del sistema, con los paneles de control. Éste cliente se comunica con Thinger vía REST y con ProceSiX vía WebSockets, mientras que entre Thinger y ProceSiX la comunicación es sobre REST APIs.

Thinger es el controlador del , MVC, el que actuliza las vistas enviadas por el propio Thinger y por ProceSiX a petición del usuario,

ProceSiX es el modelo, lógica de negocio con sus librerías y varios subsistemas que veremos más adelante. Lo importante es quedarse con que es el Modelo en MVC.

Los aspectos del arquitectura exterior son imaginativos según el contexto. Tenemos dos entidades controladas por agentes inteligentes o personas que son Thinger y ProceSiX. En el otro extremo de la conexión hay personas, sensores de todo tipo y actuadores, sistemas eléctricos, solares, baterías, huertos, calefacción , ordenadores, televisores, electrodomésticos, muebles, pantallas, persianas y cualquier otro elemento controlable desde el IoT, formado por el binomio Thinger-ProceSiX. En medio está Internet, compuesto por routers y túneles virtuales como VPN o VLANs. La arquitectura trabaja sobre TCP en el transporte aunque algunos casos será UDP junto a RTP los que lleven el transporte. SIP y la QoS serán reservados previamente a la emisión de contenidos multimedia para el control de la sesión y la tarificación por alquilar recursos de red así como los datos personales de acceso. También veremos arquitecturas punto a punto, punto a multipunto, broadcast, peer to peer e híbridas.

3.3.2- Arquitectura Interior

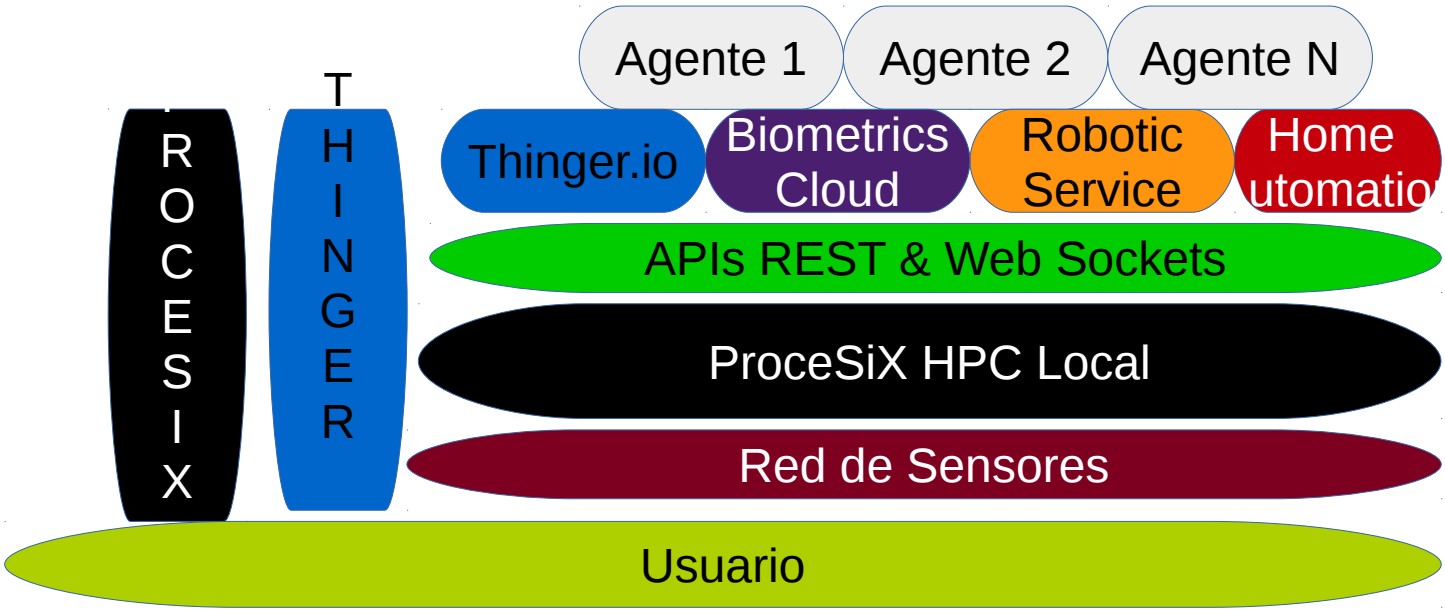
Ésta sería otra visión que alberga todos los conceptos pero que está más basada en la zona del usuario o cliente. Básicamente es lo mismo que hemos visto en el punto anterior pero dejando claro que en el cliente es dónde están todos los sensores que alimentarán de datos las aplicaciones. También tendremos instancias locales o clientes, así como instancias del sistema distribuido ProceSiX, que serán las encargados de comunicarse con su instancia padre, vertical, mediante los servicios web comentados anteriormente.

En este esquema se ve claramente que coexisten aplicaciones verticales con aplicaciones horizontales que conforman una arquitectura más robusta a la vez que flexible.

Sin embargo, pues son unos diagramas que abarcan mucho no los volveremos a ver a lo largo del proyecto ya que nos centraremos en pequeñas pruebas de concepto que siempre serán aplicables a ésta arquitectura.

Podemos decir que éste proyecto está centrado más en plantear líneas futuras que atiendan a esta arquitectura que en obtener una sola aplicación que cumpla con los esquemas vistos.

Detrás de nuestros routers casero que nos pone la operadora según nuestro contrato con ella, tenemos todos los objetos que hacen de cosa en el Internet, con aplicaciones para agricultura, industria alimentaria, industria naval, industria biomédica, espacio, industria aeronaútica, planes de gobierno, sociedad, toma de decisiones en empresas de servicios o simplemente una casa, el Internet de las Casas.



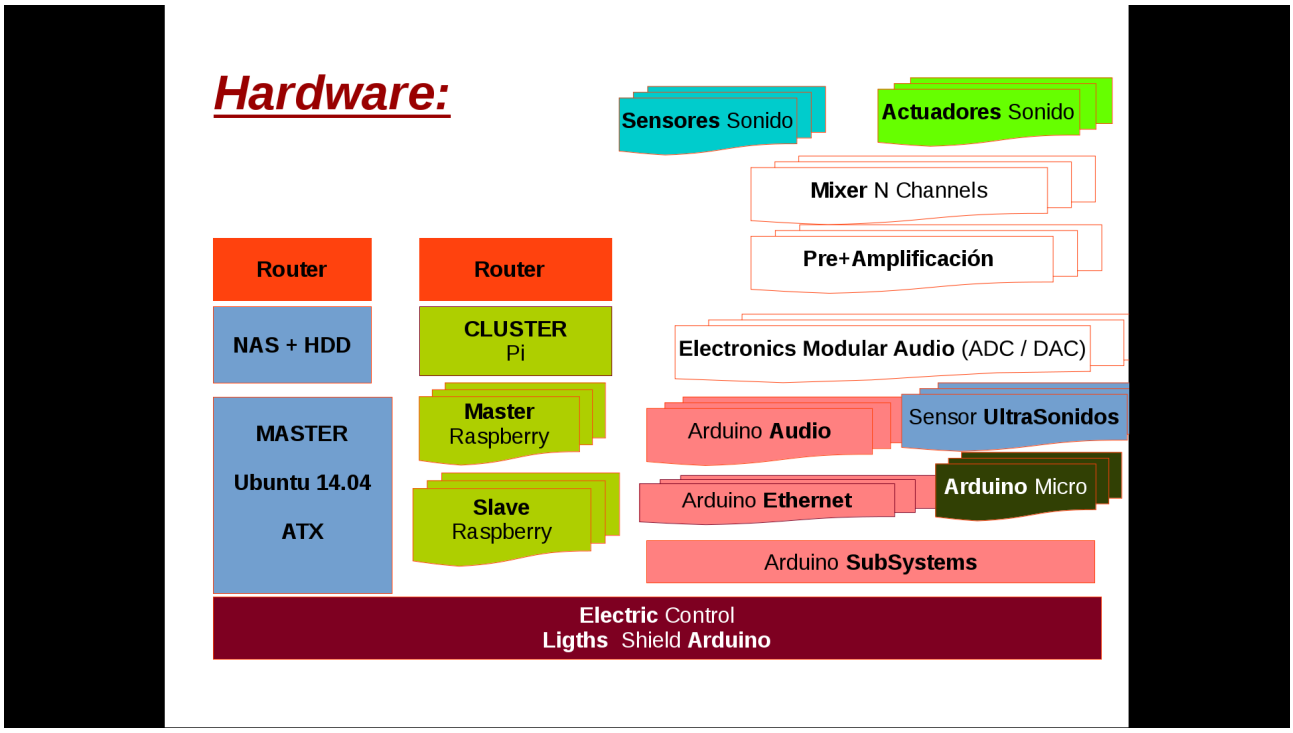
3.3.3- Arquitectura Hardware

El hardware ha sido estudiado en detalle con los datasheets u hojas de características correspondientes y además, con toda la teoría y práctica que lleva detrás la ciencia de la tecnología electrónica y de computo. Se ha soldado escudos Arduino, cables, circuitos eléctricos, circuitos de sonido y se ha prototipado sin soldar el resto de componentes.

Todos los componentes son originales y exigen un cuidado adecuado sobre todo previamente a la compra y almacenaje y uso. También hay que estar atento a la logística de los proveedores.

Lo único de la figura que aún no está implementado son los cuadros blancos rodeados por rojo, el Mezclador digital, el pre-amplificador y amplificador estéreo y el conversor AD/DA. A cambio lo tenemos integrado todo en el Arduino Micro, Ethernet y Wave Kit Arduino y una ADS1015.

En cualquier caso. Como línea futura está construir con soldadura un circuito electrónico con todas nuestras necesidades cubiertas, hecho a medida. En la bibliografía el libro “Practical Electroncis for Inventors” contiene en diferentes páginas los diferentes circuitos que estamos ya implementando a mano.



3.3.4- Arquitectura Raspberry Pi Software

Raspberry Pi ya ha sido presentada en el punto segundo. Sabemos que podemos usar Linux y Windows en la versión 2 y sólo Linux en la 1. Al usar nosotros Linux tenemos todos lo paquetes del repositorio y espejos Debian que han sido compilados para la arquitectura de las Raspberry Pi, armhf.

En las pruebas de concepto, apartado 4, y en los anexos, se detalla el software que hemos instalado en estos mini ordenadores así como su funcionalidad.

No hemos tenido mayor problema en la gestión del software con nuestras Raspberrys y hemos instalado todos los programas propios y externos con éxito y funcionales. Eso sí, seguridad no hay en exceso al estar todo en modo de pruebas.

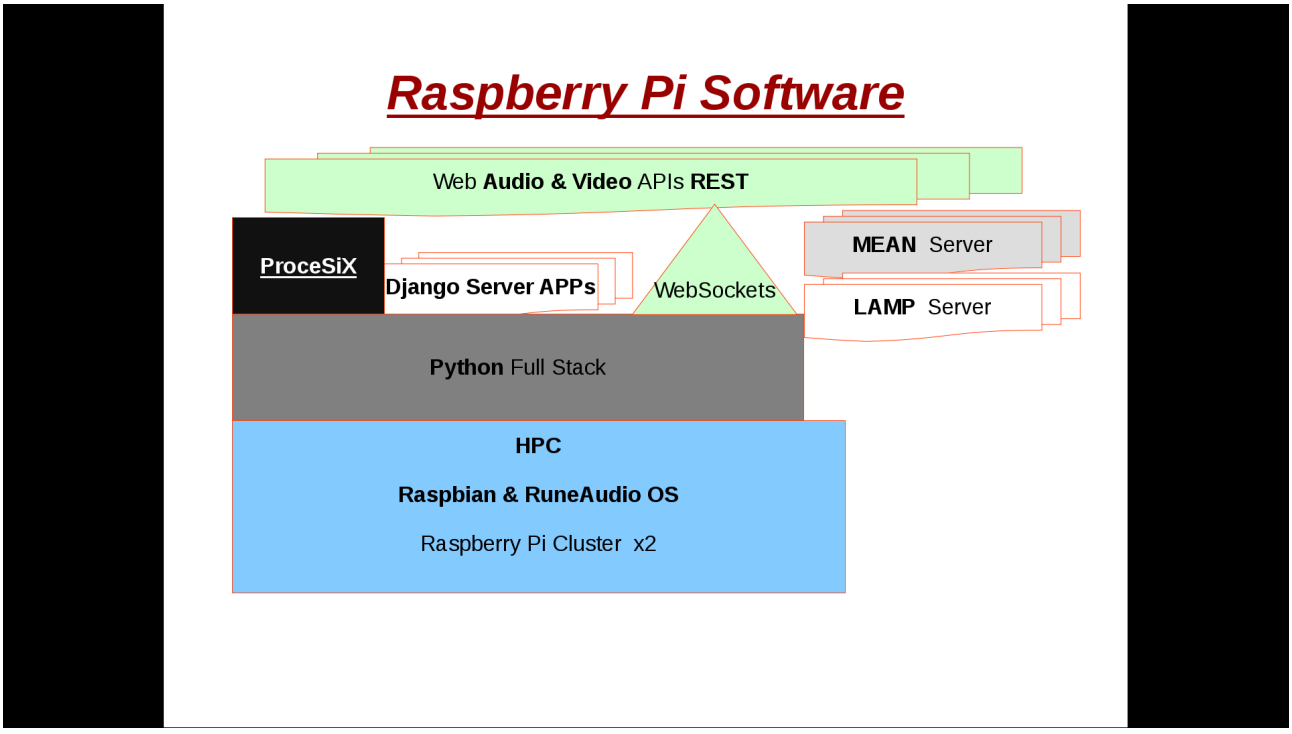
Básicamente el diagrama, está compuesto por los sistemas operativos de la raspberrys que actúan en conjunción hablando con MPI.

Después, tenemos el interprete y las librerías todo de Python, en la versión Python 2.7.

Sobre este stack de python tenemos el framework MVC Django, que es un servidor de aplicaciones como pueda ser alguno de java.

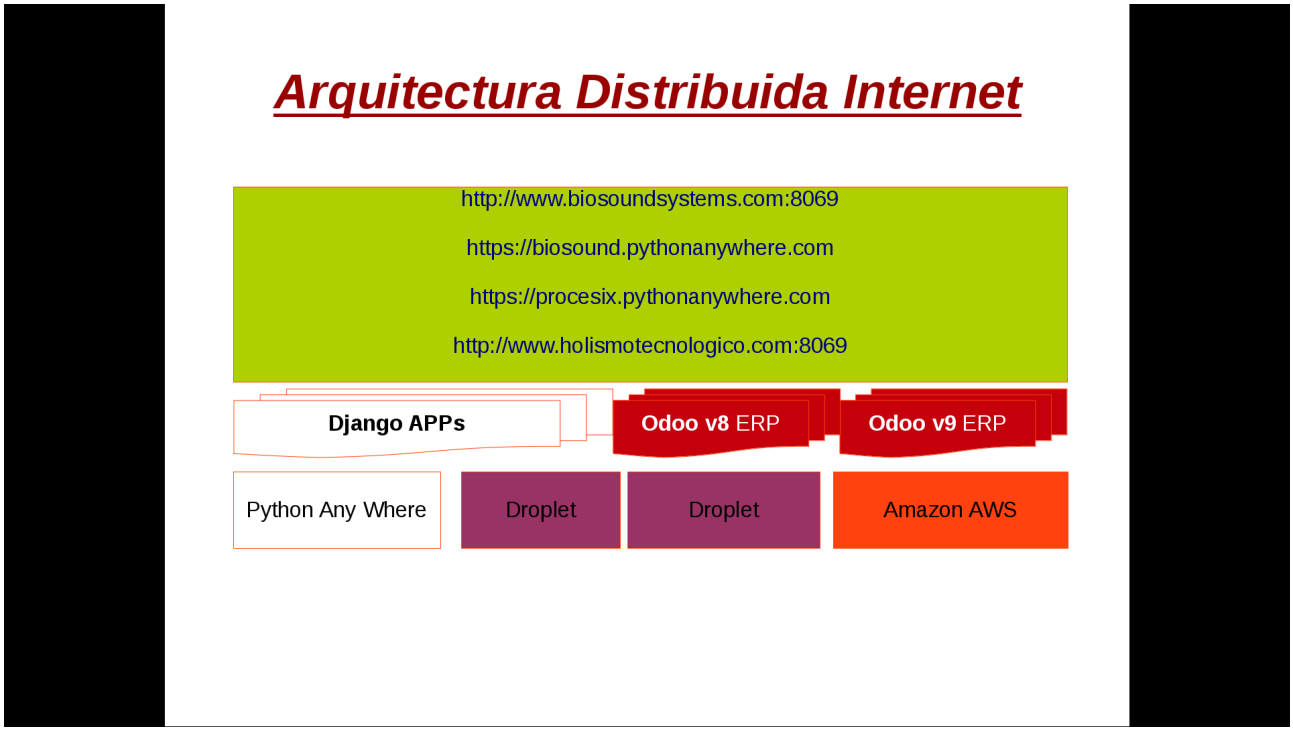
También tenemos otros dos stacks con los que crear recursos, estos son LAMP (Apache) y MEAN (Javascript).

Todos ellos irán integrados con REST y WebSockets para aplicaciones de Audio y Vídeo



3.3.5- Arquitectura Distribuida Internet

La distribución de software en internet es un paso esencial para cualquier desarrollador que quiera propagar su software, sus escrituras digitales como programador. La aplicación estándar para guardar, gestionar, publicar y mostrar ese software es github, el rey para los programadores. Hay muchos lenguajes, máquinas virtuales, sistemas operativos, hiper-visores, navegadores web, sistemas operativos móviles, sistemas operativos robóticos y sobre todo muchísimo software para la ingeniería de reutilización o reciclaje del software funcional, ya que la obsolescencia programada es inmediata si se trata de software, que cada día se actualiza y cambia. El software en sí se puede propagar por métodos virales que se comportan como el cáncer cuándo crece, se propaga a una velocidad de vértigo. Estos software son los gusanos, los troyanos y en general los virus informáticos que se propagan de manera inteligente para alcanzar a sus objetivos o víctimas.



Nuestro caso es diferente ya que nuestro objetivo principal es llegar al fondo de la comunidad, por lo que distribuiremos el software libre de descarga y uso con métodos de distribución más modernos que los hipervisores como es docker con sus droplets.

Otra tecnología punta es toda la arquitectura de Amazon desplegada para dar servicio completo a todo el globo. Además han añadido una plataforma para el Internet de las Cosas que hace la competencia dura a otras plataformas como las nuestras.

En estas infraestructuras ofrecemos como SaaS, Odoo v8 y v9, un completo sistema para la gestión y administración completa de la empresa, ERP, CRM, CMS …

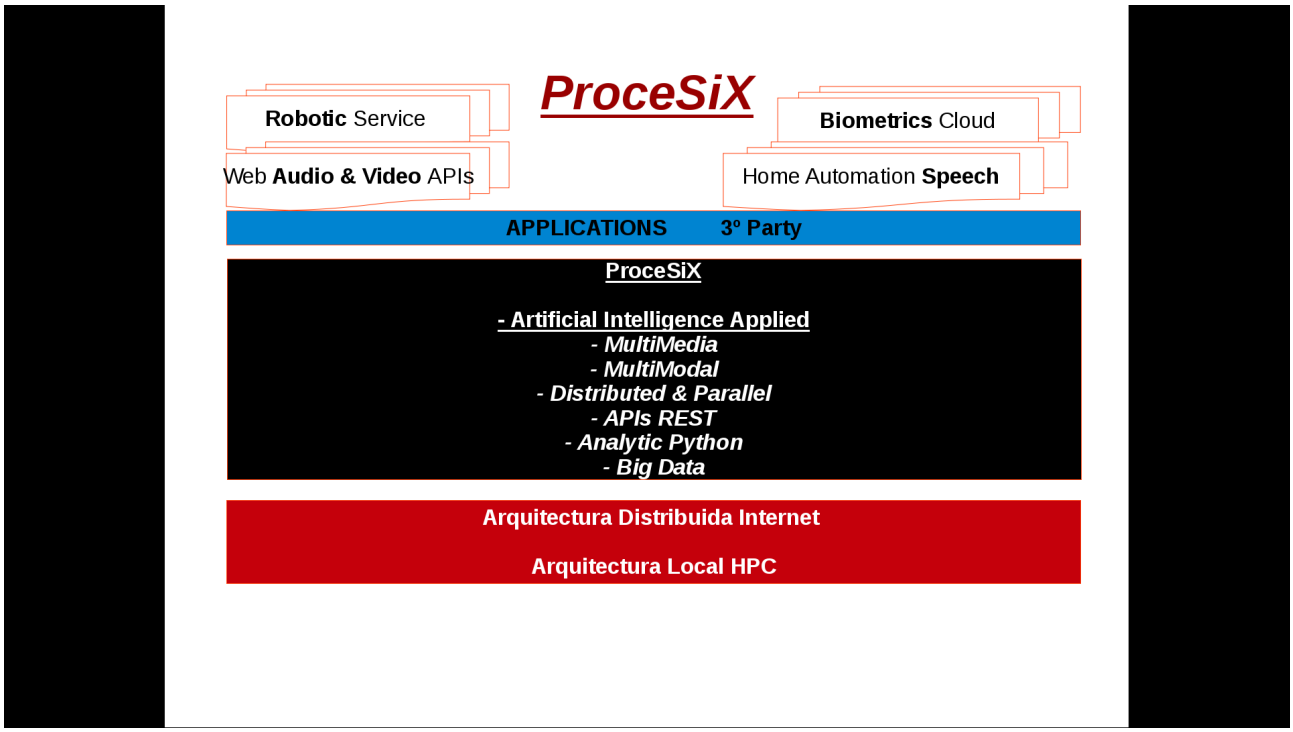
Y por supuesto, también está la posibilidad de crear aplicaciones web con Django y con todas las que ya hay hechas, ofreciendolo como SaaS. La principal será el hilo de seguridad , el control de los servidores con SSH, el API REST y el servidor WebSockets.

3.3.6- Arquitectura ProceSiX

ProceSiX, software para Inteligencia artificial en Internet de las Cosas, distribuido en Local mediante hardware abierto y en remoto mediante servidores alquilados, aporta una pieza esencial en entornos analíticos como todos los mostrados en las aplicaciones explicadas.

Como es software Open Source, se podrán descargar de la web las fuentes, la documentación, los ejemplos, y la información multimedia asociada a la versión beta y a la alfa u oficial. Serán distribuidas en discos SSD de 320 GB o en varios DVD abarcando todas las versiones, pruebas, ejemplos, software, fotografías, vídeos, listado de paquetes a instalar, scripts, etc...

La tendencia y evolución que va a perseguir ProceSiX es dar sentido a conceptos dentro del Internet delas Cosas pero separados en algunos aspectos. ProceSix ofrecerá una arquitectura para Big Data, con Inteligencia de Cómputo Distribuida, con Sistemas Multiagente y tendencias hacia modelos de Ecosistemas Vivos, Vida Artificial. Al estar el núcleo de la aplicación presentado como un panal de abejas al modo de las células de las redes móviles, tendrá comportamientos grupales o sociales asociados en el modo de funcionar, que puede ser cooperativo con el usuario o competitivo con el usuario o con otro ecosistema o red, dando lugar a las teorías evolutivas de lo biólogos y científicos en las T.I.C.s.



Como vemos en la imagen, la base arquitectónica de ProceSiX es un cluster HPC Local (Armario) y un grupo de servidores actuando en conjunción distribuidos a través de Internet.

Después, como deciamos, ProceSiX evolucionará según los planes de hacer de ella una plataforma para Inteligencia Artificial aplicada, Multimedia y Multimodal, Distribuida y Paralela,

con capacidad para Big Data y Ciencia de Datos y todo ello desembocando en un sistema de APIs JSON y WebSockets para interactuar con otros servicios, como Thinger o BioID.

Aplicaciones que ya están incluidas en las líneas futuras y que son abarcadas por alguna prueba de concepto son las que vemos en blanco en la imagen. Biometria en la nube, Asistencia con Robótica de Servicios, Edificios Inteligentes Automatizados o tratamiento de Audio y Vídeo en la Web y en el sistema operativo Ubuntu.

El objetivo es que pueda haber un kit de armario o mueble robotizado e informatizado en los salones, oficinas, edificios, bares, discotecas, teatros, salas de conciertos, platós de televisión, cocinas, servicios públicos, cuartos de contadores, plantas energéticas de todos los lugares del planeta y a un bajo coste de producción y de consumo posterior, pudiendo incluso hasta generar beneficios brutos en electricidad, reputación on-line, imagen de comunidad, comunicación y marketing, CRM, ventas, compras, almacén y todas las divisiones como las de la gestión de la organización para una empresa de I+D+i.

3.3.7- Arquitectura Thinger

Thinger es la otra parte del binomio para el Internet de las Cosas, junto a ProceSiX. Pero Thinger es la plataforma de monitorización, comando y control de manera vertical a todas las capas de TCP/IP y con interfaces REST y WebSockets que consumen y pueden ser consumidas.

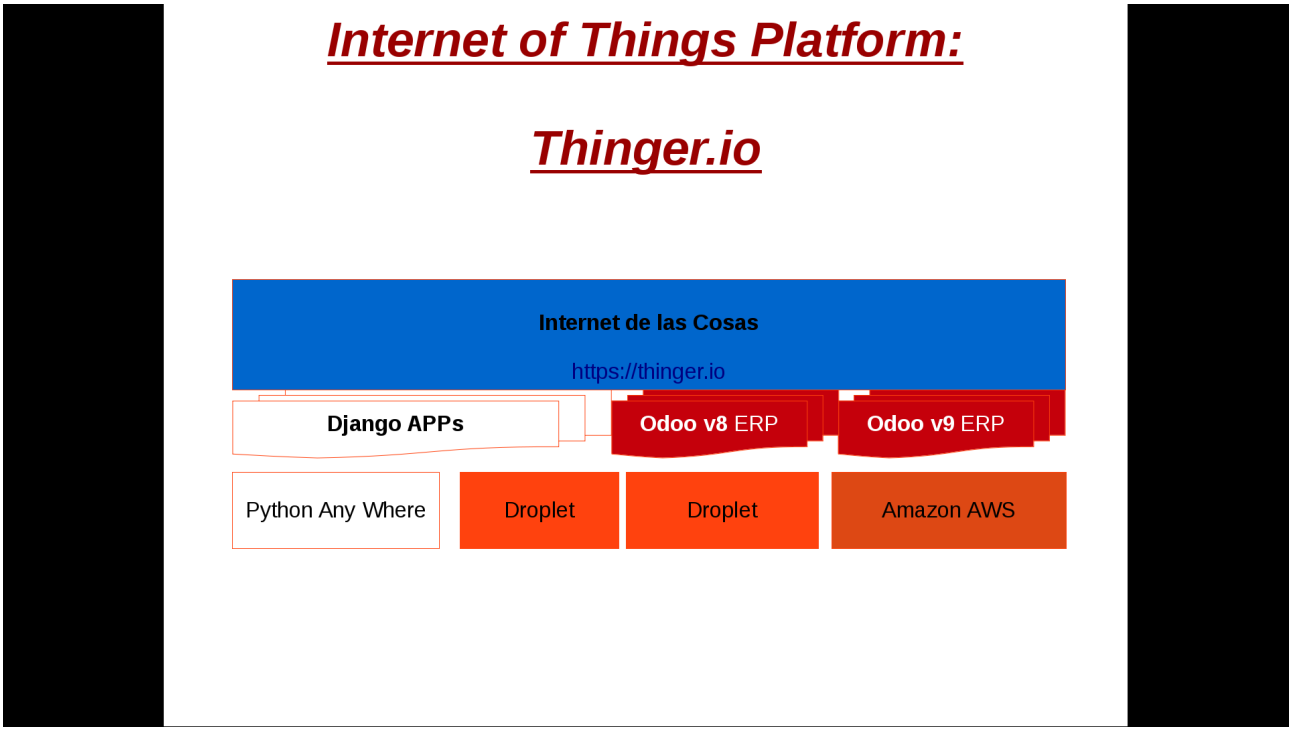
Thinger es sencillo de instalar en una Raspberry Pi, en Intel Edison, en Linux, en Mac OS, y en Arduino y nosotros no hemos tenido problema con la instalación y manejo del software porque tiene la suficiente documentación y además una gran lista de ejemplos, que es de la que se aprende.

Es una plataforma para el Internet de las Cosas, pero en vez de para añadir Inteligencia Artificial con Big Data a las Cosas de Internet, añade una interfaz sencilla para la monitorización de los dispositivos y para su interacción mediante abstracciones API editando únicamente su clase Main.cpp.

Rápida, escalable, segura, flexible, eficiente, altamente optimizada, protocolos internos propios para la programación de alto rendimiento o intensiva, esto es Thinger.

Está escrito en C++ con librerías también de C e incluso librerías propias, pero más específicamente, está escrito en el estándar C11.

Internet está de enhorabuena con esta plataforma: ¡¡¡ Internet of Thinger !!!



En la imagen podemos ver que nuestra integración con Thinger es transparente. Y en el

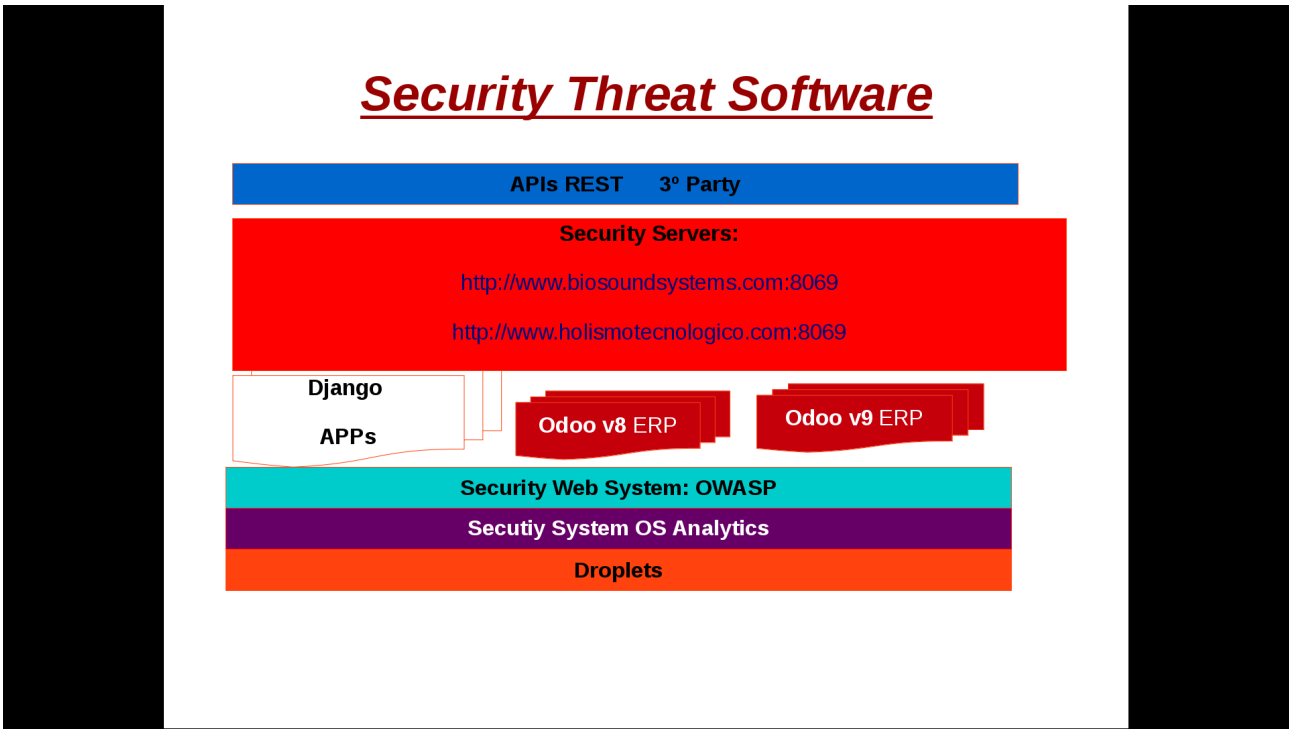
punto 2 hemos presentado el ejemplo particular de la arquitectura de Thinger, que es propietaria.

3.3.8- Arquitectura Seguridad

La seguridad tienen millones de aspectos o caras de una misma moneda. Existen sofisticados tipos de ataques dirigidos a una sola persona u organización y según los datos oficiales de las entidades reguladoras de guerra y paz en el ciberespacio cada vez se expande más el crimen organizado con Internet como intermediario, aunque ojo que puede haber ataque en persona como siempre ha ocurrido.

Hemos diseñado desde el principio el sistema orientándolo a la seguridad. Podríamos haber utilizado OpenBSD, pero hemos comprobado que es algo complejo que requiere mucha dedicación de recursos y tiempo aunque al final sea en sí más seguro. Ubuntu es nuestra solución, junto con Debian y Raspbian y RuneAudio OS así como Arduino, que no olvidemos también puede ser víctima de algún ataque informático, a propósito o aleatoriamente, así todos somos objetivos probables en una u otra medida.

Hemos intentado diseñar ProceSiX para que cumpla con el estándar de seguridad OWASP, con el PEP 8 de Python y buenas prácticas en general, siendo el objetivo que el pentesting y la explotación sea un proceso continuo en el tiempo alimentado así nuestra base de datos de big data con Spark embebido en ProceSiX.



Nos hemos encontrado con que nuestra plataforma de pruebas es bastante vulnerable comparada con Thinger o con Odoo, pero como línea futura está planteada como una fuerte parte del desarrollo, sobre todo que todo el ciclo cumpla los estándares de desarrollo seguro y funcional.

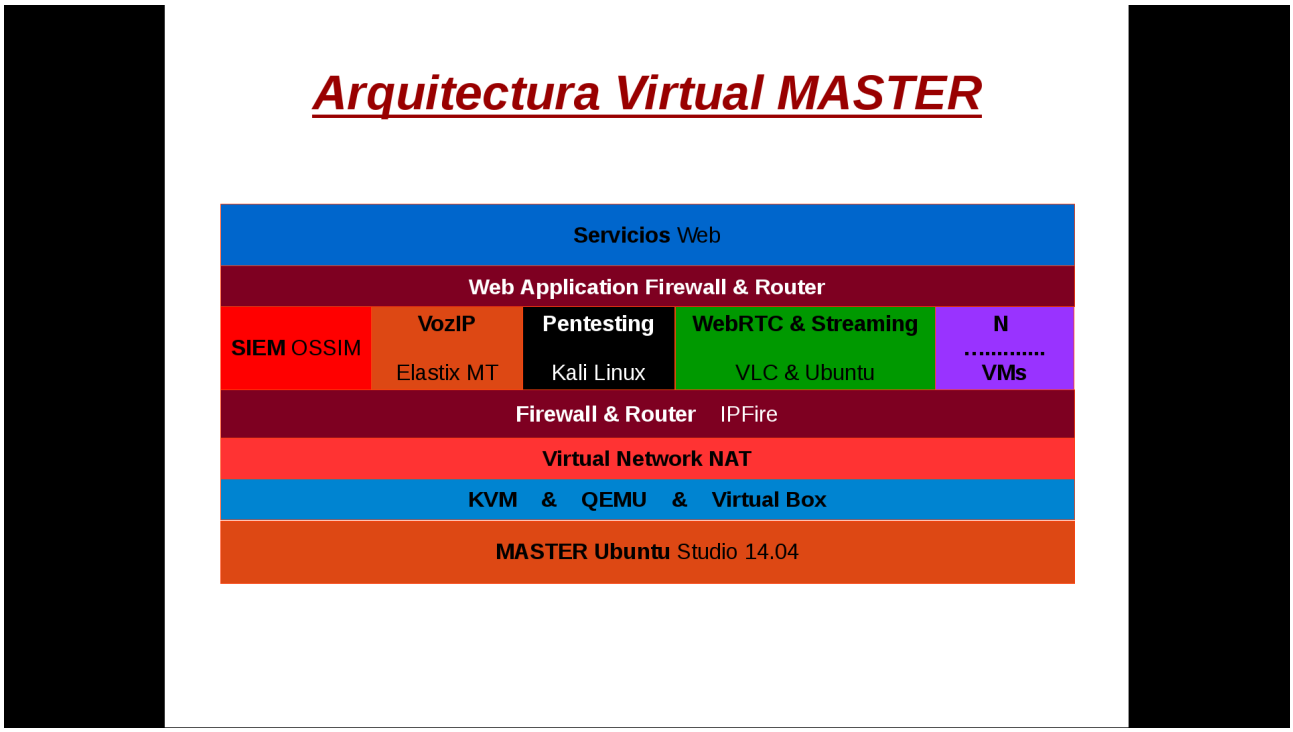
Por lo tanto, hemos realizado ataques a Raspberry pi, Ubuntu Studio, Thinger, Digital Ocean, OVH, y Python Any Where como representante de Amazon. Lo hemos hecho tanto a nivel web como de sistema operativo y resulta que los dispositivos más vulnerables o con más riesgo por la cantidad de puertos que tienen abiertos o filtrados son los routers, la mayor superficie de exposición sería del router para fuera a través del modem, nuestra IP pública podrá ser monitorizada con software de vigilancia tecnológica implementado por debajo como un ataque continuo o pentesting continuo.

3.3.9- Arquitectura Virtual MASTER

MASTER es el ordenador con más recursos del armario con ruedas. Es el mayor responsable de toda la infraestructura, tanto hardware a medida como software a medida. Ubuntu Studio Low Latency es el sistema elegido como Host para el MASTER. También tenemos dos hiper-visores que tienen máquinas virtuales alojadas en huésped.

Las máquinas virtuales son una red NAT, un Router y Firewall, IPFire, los dos hiper-visores(KVM y Virtual Box), el SIEM OSSIM, el Elastix MultiTier, la estación de vigilancia tecnológica con Kali Linux, un Ubuntu Server como servidor de Streming con VLC.

Y, sin virtualizar, tenemos el sistema operativo sobre el kernel de baja latencia de Ubuntu Studio y con todo el Software de Edición Multimedia Profesional Open Source que conforma lo que llamamos el Open Multimedia Work Station que veremos a continuación.



3.3.10- Arquitectura Open MultiMedia Work Station

Físicamente tenemos que convertir la señal de analógico a digital con tarjetas de adquisición de audio y de vídeo analógico y digital o bien tarjetas de captura multimedia.

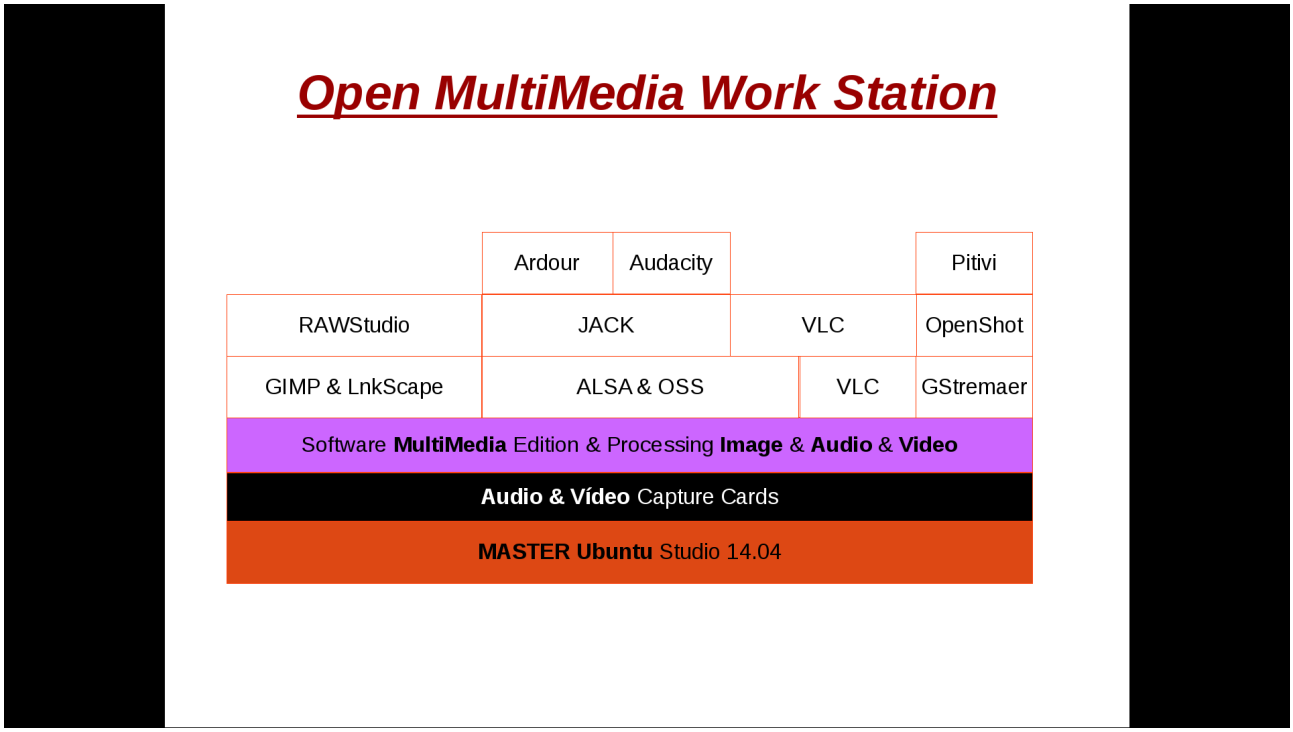
Para hacer esta estación disponemos de un repositorio enorme de diferentes opciones para elegir entre un software u otro de procesado de audio, vídeo o imagen y diseño y publicidad, iluminación y animación 3D.

Recordemos que el que hemos elegido nosotros es: RAW Studio, GIMP, LnkScape, ALSA, OSS, JACK, VLC, Ardour, Audacity, Pitivi, OpenShot y Gstreamer de Fluendo.

Con este elenco de software libre y la formación adecuada nos podemos hacer una estación de tratamiento de audio digital en nuestro propio ordenador personal, pero también de vídeo, fotografía, publicidad, diseño, iluminación y animación 3D.

Lo que más nos ha interesado del proyecto, a parte del péndulo, es el Sistema de Baja Latencia de Streaming de Audio 3D para un Sistema de Grabación Multipista, explicado en un articulo de la bibliografía.

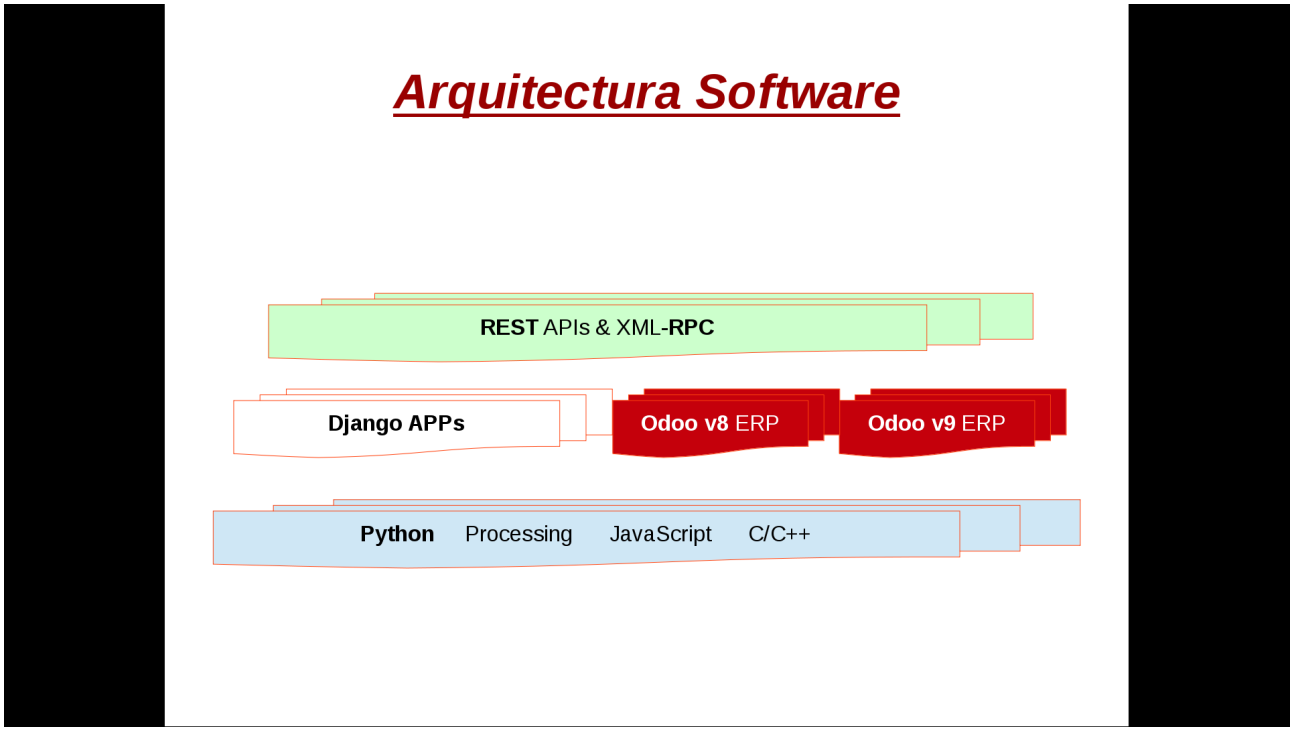
Sí, con esta estación podemos adquirir, mezclar, pre y post produción de todo tipo de multimedia, también podremos emitirlo a la red en la que si reservamos recursos nos dará suficiente prioridad en nuestra comunicación como para que se a efectiva, útil y práctica.



3.3.11- Arquitectura Software

El software no nos ha dado mayor problema que el de elegir librerías, lenguajes, métodos, funciones, objetos, parámetros, datos, clases y aspectos para programar las pequeñas pruebas de concepto que hemos preparado.

Ya que el desarrollo de software es mas una línea futura que presente, no nos hemos enfrentado a problemas muy complejos en la práctica, sólo hemos visto la teoría y un poquito de práctica para hacer la demostración.



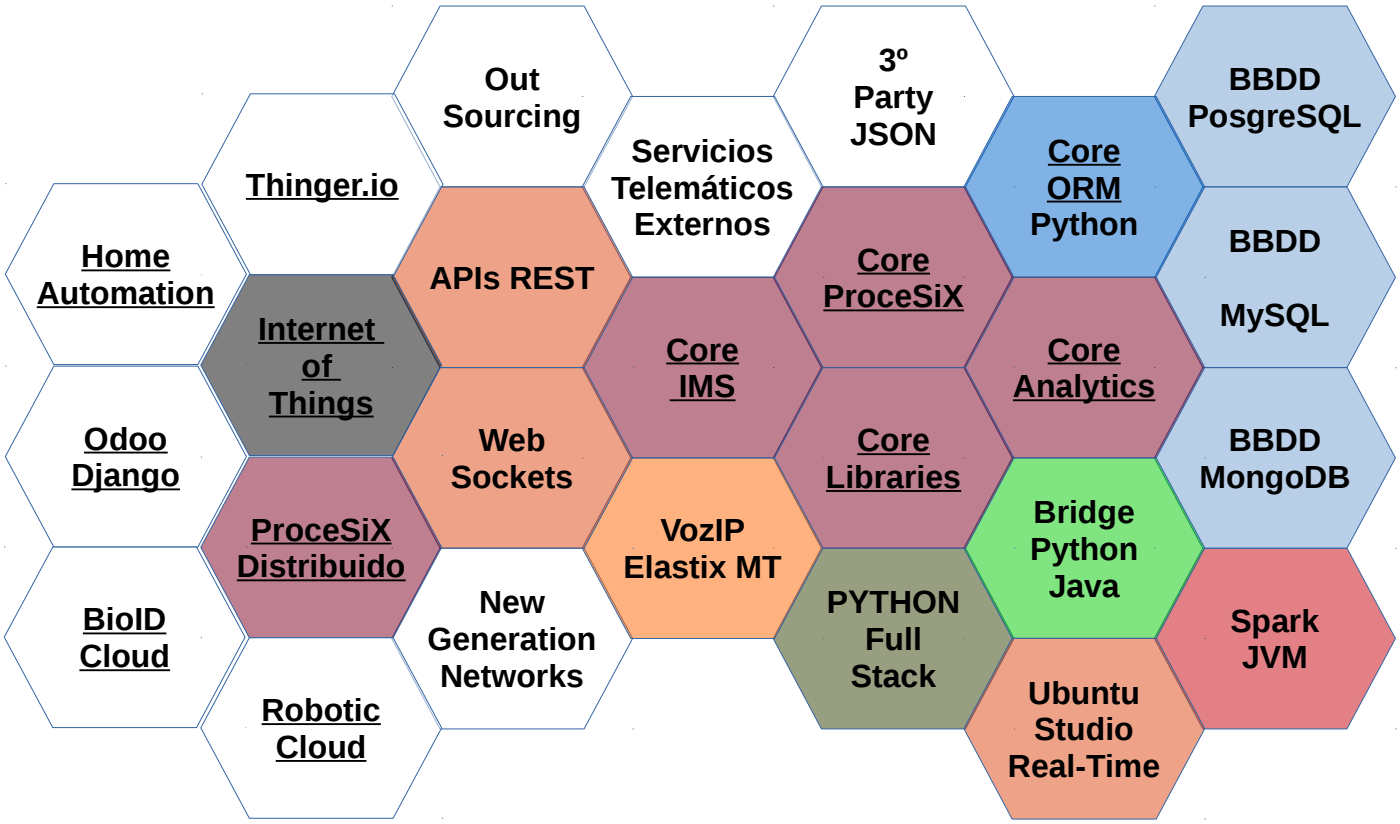
3.3.12- Arquitectura Bio-Inspirada

La inteligencia en enjambre es una estructura de agrupación que funciona bien en sistemas probados como los panales de abejas o los artificiales como las redes móviles.

Hemos estructurado nuestra aplicación de este modo ya que estamos muy motivados con las aplicaciones y soluciones que puede aportar la Vida Artificial a la Inteligencia Artificial en un vasto terreno de posibles aplicaciones, todo el globo conectado con cosas.

Espero que después en la práctica sea una buena estructura que funcione bien y que sea eficiente y segura. Pero hasta que no se haga un desarrollo formal intensivo no podremos enlazar cada molécula de información y conocimiento.

Valoramos algoritmos de inteligencia en grupo como los basados en las mismas abejas, en hormigas, en manadas, en bancos de peces así como en individuos aislados integrados de alguna manera en su hábitat.



Lo principal es el núcleo o core que está compuesto por las librerías para ciencia de datos y para análisis automático. Por un núcleo de IMS (IP Multimedia Subsystem) implemntado en Java. Y el núcleo en sí de la aplicación llamado ProceSiX y escrito en Python.

A la derecha de la imagen tenemos el sistema de bases de datos insertado en el core mediante un capa ORM escrita en Python. Tenemos varios sistemas de bases de datos que conforman un Data Ware Hause (DWH) en la prática.

Todo el núcleo se ejecuta sobre Ubuntu Studio con el kernel de baja Latencia para multimedia y el resto de ProceSiX se ejecuta bien en Python y Django o bien mediante un puente de Python a Java para aplicaciones Java. Como ProceSiX es el core estará escrito en Python y optimizado en C/C++.

Después tenemos las máquina virtuales, entre las que destaca Elastix MT para comunicaciones unificadas y WebRTC que funciona sobre redes de nueva generación.

Mediante los WebSockets y los APIs REST interactuamos con Internet o Things, con Thinger con la versión distribuida en Internet de ProceSiX, y sobre otras aplicaciones comentadas como Home Autmomation, BioID, Robotic Services, y Odoo junto a Django con diversos tipos de aplicaciones.

4- Pruebas de Concepto

Taller:

4.1- Interfaz Acústica

Hay que tener en cuenta que el contexto de esta interfaz además de acústica es electroacústica e informática.

La prueba de concepto inicial se ha convertido en un conjunto de instrumentos virtuales comandados por la guitarra española. Son reales, pero lo que digitalizan bien se podría llamar VST y ser plugin de Audacity por ejemplo o embebido en la web como hemos visto.

Por lo tanto, hemos creado principalmente una guitarra electroacústica con 6 sensores de sonido y uno de ultrasonido, un micro arduino, un escudo arduino ethernet y otro de audio, un vúmetro con diodos y todo ello conectado a los routers y a la raspberry pi 2 con su cámara de vídeo, atornillados en la base de la guitarra. Todas las demás piezas están también a mano de destornillador por lo que se puede convertir un instrumento en electroacústico en cuestión de instantes, siempre que sea de madera, como es nuestro caso.

El armario se ha quedado sin micrófonos, y sólo tiene la Raspberry Pi B, los sensores de humedad y temperatura analógicos, así como con leds para un semáforo rojo y verde y los altavoces como actuadores. Además tenemos el hub USB, una cámara de vídeo, un radio despertador, tres bombillas con interruptores automáticos o relés y manuales como los clásicos. Y siempre, ayudados estéticamente con la construcción que da aspecto de juego que es un pack MOLTO para mayores de un año de edad y un pack LEGO Classic. Lo más destacable es que el semáforo será quién permita el paso o no en un supuesto control de accesos comandado por la vozde cada persona.

Esta prueba de concepto está compuesta por un sistema de sonido básico que es capaz de percibir sonido a partir de cierto umbral y actuar en consecuencia, en nuestro caso encendiendo diodos luminosos, el vúmetro.

Tenemos 7 espectros brutos, 6 en el rango del sonido y 1 en el rango de los ultrasonidos. Con los primeros tendremos varias señales acústicas que los sensores de sonido transformarán en un pequeño nivel de voltaje que transformaremos linealmente a un número arbitrario. Este número arbitrario será el lanzador de una escala dividida en 8 umbrales que será nuestro dispositivo de monitorización, un vúmetro, compuesto por diodos LEDs. Con los ultrasonidos, igualmente leeremos voltajes que transformaremos en una unidad de medida de la distancia gracias a que el sensor integrado tiene un lanzador y un capturador de eco. Según se envíen o reciban señales se encenderán uno u otro diodo LED.

A parte de la entrada de sonido o micrófono, tenemos dos salidas en forma de altavoces que es capaz de emitir tonos puros durante un tiempo dado dando lugar a posibilidades como “dar la voz de alarma” o informar en forma de sonido sobre el estado del sistema. Así se puede establecer un código para poder interpretar la información sonora tal y como hacen los zumbadores de los ordenadores personales, PC. También se podrá reproducir y grabar audio digital desde la Raspberry Pi, como comentamos a continuación.

En cuanto a reproducción de sonido, también tenemos la posibilidad de hacerlo mediante el puerto HDMI de la Raspberry hacia una pantalla con altavoces, o el propio Jack de 4 pines que también soporta un micrófono con la tarjeta de sonido previa adecuada.

Basado en Arduino hemos soldado un kit de audio que conforma un escudo compatible con el sistema Arduino. Con él tenemos un reproductor compacto de audio en formato bruto o wav que podremos conectar directamente al jack de los altavoces de nuestro prototipo. También disponemos de un reproductor MP4 como reproductor.

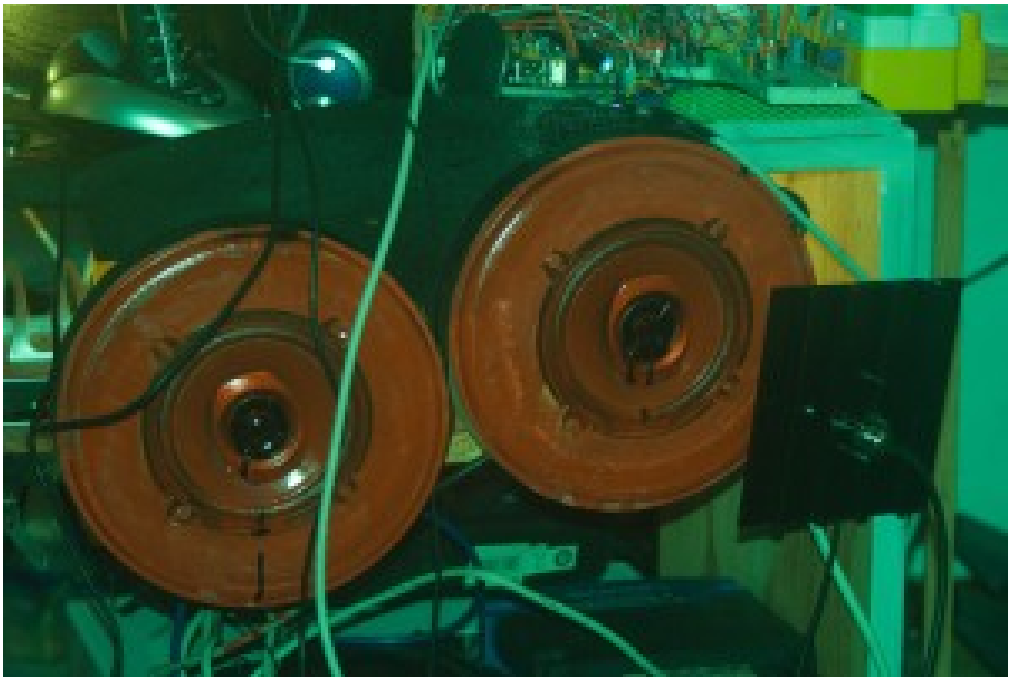
Con este conjunto de dispositivos podemos ser capaces de plantear un sistema de control de sonido, audio y voz analógico y digital. Podremos llamarlo Estación de Audio Virtual de Trabajo. Con voz funcionará como una interfaz de comando y control, o de identificación biométrica. Con audio y música será capaz desde la adquisición hasta la grabación y reproducción.

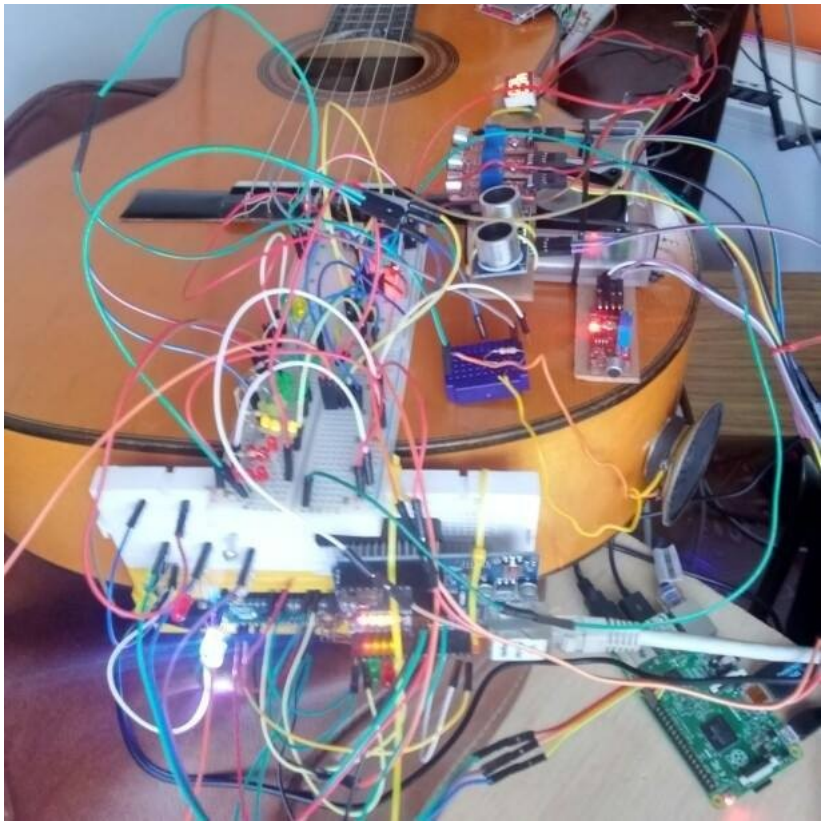
El sonido bruto capturado nos servirá para establecer umbrales de altura que digitalizaremos para poder codificarlos en función de la aplicación que se va a nutrir de estos datos.

Mediante la reproducción de sonido cerraremos la interfaz de audio dando realimentación al usuario, aunque nuestra maqueta será capaz de hacer de hilo musical en una casa o de equipo de sonido en un automóvil así como de enlace por WiFi para las conversaciones y música de uno o varios teléfonos móviles o instrumentos electroacústicos.

Como podemos ver, ésta interfaz acústica no nos ofrece una aplicación concreta sino que conforma varias pruebas de concepto que serán aplicables a muchos tipos de aplicaciones. Como a lo largo de todo el proyecto, vemos que se cumple el objetivo de plantear diferentes esquemas que hagan las veces de cimientos para múltiples aplicaciones, pero a la vez, para ninguno en concreto.

La aplicación más importante del proyecto forma parte de la interfaz de percepción de sonido. Es un instrumento virtual, una guitarra española con transductores para digitalizar el audio y procesarlo además de transmitirlo por la red, con raspberry pi y ubuntu studio de 64 bits en red. También podemos, obviamente capturar las señales de voz, armónica y de un cajón flamenco o de cualquier otro elemento electroacústico. Todo junto lo enviaremos por WebRTC para que varios usuarios puedan compartir flujos multimedia. Así podrán ensayar cómo si fuera realidad virtual o aumentada aplicada a la música.





4.2- Interfaz Lumínica

La interfaz lumínica está compuesta únicamente por actuadores. Pero una vez llegados a este punto, sería inmediato añadir un sensor lumínico que nos diera como valor el número de lúmenes en cada instante muy interesante para diferentes aplicaciones, siendo la más directa utilizar esta información para controlar una estación de energía solar.

A diferencia de la interfaz acústica, en esta prueba se ha implementado una aplicación web desde donde se pueden controlar los canales compuestos por diodos LED y los compuestos por varios conmutadores automáticos o relés que encienden o apagan bombillas enchufadas a la red eléctrica.

También nos ofrece realimentación de naturaleza diferente a la que representa. Por ejemplo, el vúmetro mide sonido pero tiene una interfaz visual basada en código de colores.
Y, con el escudo de relés, se puede hacer incandescer varias bombillas a la frecuencia deseada, por ejemplo, la de la música que suene en cada instante en vez de los 50 Hz de la red eléctrica española.

Como a lo largo del proyecto, este ejemplo sirve para ilustrar las posibilidades de control remoto que nos ofrece la plataforma, siendo este un claro ejemplo de automatización domótica. Tampoco es una aplicación concreta, pero repetimos que planta la semilla para insertar lógicas más sofisticadas para por ejemplo control y emulación de presencia o sistemas de iluminación inteligente para salas de baile o hasta simplemente control y ahorro del consumo energético gracias a que los relés pueden estar incluidos en el cuadro eléctrico de la casa y así tener automatizado cualquier elemento eléctrico de la cocina, el servicio o del terreno exterior que quizás pueda tener un pequeño huerto con irrigación automática.

4.3- Interfaz Visual

Este ejemplo se sirve de dos cámaras para monitorizar en remoto el entorno que éstas capturan y como en el resto del proyecto ésta información está accesible desde la web; ya sea desde un ordenador, tableta o teléfono móvil con acceso a redes de datos.

Por lo tanto, tenemos una visión del entorno próximo que capturen las cámaras que posteriormente se procesa para detectar el movimiento y lo que se muestra es sólo éste. Además, se pueden implementar sistemas más complejos de cámaras que ofrecen una valiosa información para su análisis, procesado, grabación, reproducción y transmisión. Ésta será una importante linea futura.

Este ejemplo sirve para ilustrar lo sencillo que es establecer un sistema de monitorización visual, vídeo conferencia, videovigilancia, etc... Con la plataforma presentada hasta el momento.

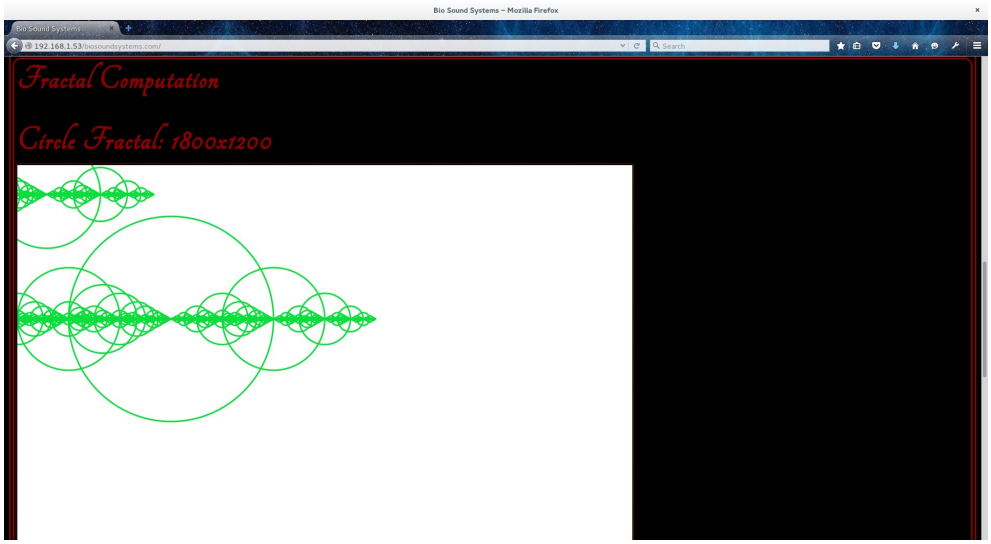
4.4- Interfaz de Control y Monitorización Web

Debido a que no buscamos realizar una aplicación con una interfaz web integral, hemos subdividido en diferentes interfaces, servidores web y servicios web cada prueba de concepto.

La interfaz más completa está servida por una pila LAMP bien conocida. En ella tenemos un reproductor de audio con sus controles asociados, en HTML5. Además, con el Web Audio API hemos implementado en el cliente web una lógica que reproduce tonos puros durante los primeros segundos de carga de la página, demostrando las posibilidades de procesamiento de audio que ofrecen los navegadores modernos. Podremos hasta implementar una mesa de mezclas con javascript.

Así, con la tecnología actual de los navegadores web tenemos acceso a muchas API para ejecutar cualquier programa. Nosotros hemos utilizado HTML5 y CSS3 así como JavaScript y Processing en cliente y Python y PHP en servidor.

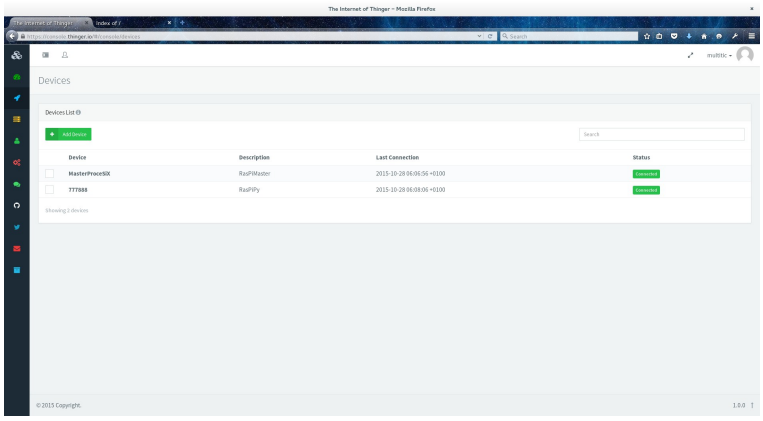
Además, en éste caso con Processing, hemos implementado un algoritmo recursivo que dibuja círculos semejantes con una naturaleza de expansión fractal. Vemos que éste programa es el que más carga demanda al navegador. Es un ejemplo con el que demostrar el uso de otra tecnología como Processing embebido en la web. Los fractales también podrían ser utilizados para realizar codificación de vídeo en el mismo navegador, por ejemplo multicapa. Otro ejemplo de algoritmo complejo es el famoso Juego de la Vida de Conway basado en autómatas celulares, embeberlo en el navegador con Processing es inmediato. Queda para linea futura.



También embebido en ésta interfaz tenemos el resultado de los movimientos capturados por las cámaras. Pero este caso es un programa muy completo, motion, con servidor propio que funciona en un puerto diferente al apache aunque se muestre en la misma interfaz.

También en otro puerto tenemos dos servidores y frameworks web, bottle y django, con una interfaz y lógica para controlar las luces de la maqueta mediante botones como hemos comentado en el apartado de interfaz lumínica el primero, y con django la lógica para crear la interfaz del API REST y WebSockets.

Por otro lado, en un servidor remoto en Internet, tenemos la interfaz web de control y monitorización de la plataforma para el Internet de las Cosas, Thinger. Mediante ésta interfaz tenemos monitorizados los elementos que hayamos configurado como es el caso de las Raspberrys, los Arduinos e incluso el otro servidor remoto nuestro en el que se ejecutan tanto el ERP como los programas de análisis de la seguridad como otro servidor Django que hace las veces de aplicación REST y WebSockets.



Todas estas “cosas” interactúan con el núcleo de ProceSiX y con el servidor de Thinger formando un binomio capaz de implementar aplicaciones verticales y horizontales de cualquier tipo, en nuestro caso orientado a la multimedia.

Otro sistema operativo, a parte de Raspbian, llamado Rune Audio OS está especializado en audio y puede ser utilizado para hacer streaming en casa o en internet con la conexión de red adecuada así como cadenas de proxys que hacen de buffer para cumplir los requisitos de tiempo impuestos por las aplicaciones multimedia. Este sistema operativo no tiene interfaz de ventanas, sino que tiene un servidor web integrado que sirve un GUI Web con la que el usuario interactúa, y desde la que puede gestionar toda su música y dispositivos de almacenamiento.

En el caso de Raspbian Jessie, la última versión de Debian para armhf en la Raspberry Pi, tenemos las mismas pilas de protocolos con sus respectivas aplicaciones como son Django y Python, Javascript y Nodejs, Apache y PHP5 siendo un LAMP clásico, Python integrado con C/C++ y Processing embebido como script en las páginas servidas por los servidores. En la versión 2 de la Raspberry Pi todos estos servidores tienen muchos mas recursos disponibles para consumir y al final al desembocar todo el núcleo en un API REST y WebScockets, también son recursos que serán consumidos por los diferentes usuarios.

4.5-Integración Servicios Web

Para integrar cualquier servicio con nuestra plataforma tenemos básicamente las dos opciones vistas que son SOAP o REST así como Websockets para flujos multimedia.

Hemos elegido REST porque creemos que es la manera más sencilla además de la que más se está extendiendo actualmente con su orientación al recurso (resource) o al dato, especialmente con JSON como formato de los objetos intercambiados entre las aplicaciones Web.

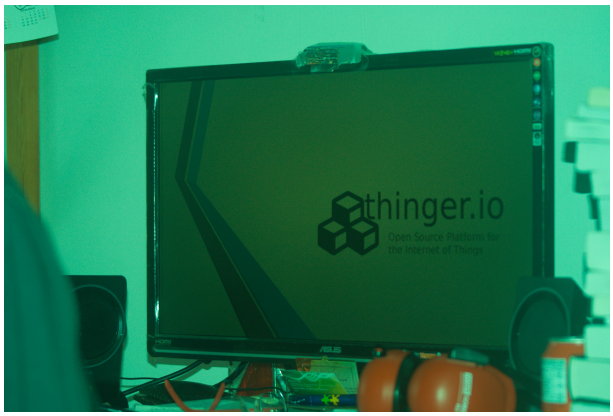
Por lo tanto, para implementar estas integraciones la mejor solución es desplegar una API REST capaz tanto de consumir de otras APIs como de ser consumida en el caso de que nuestro servicio sea de interés de otra tercera plataforma.

Hemos utilizado el Django-Framework-Rest que es una aplicación que nos soluciona o abstrae todos los detalles de implementación del sistema de APIs. En el anexo correspondiente vemos que instalar como APP de Django el Framework REST es una tarea sencilla y que nos da como resultado un API con derechos de acceso o autenticación para el administrador y otros usuarios y así éstos puedan ver y llamar los métodos disponibles en la API e incluso embeberlos en sus respectivas aplicaciones. Tendrán nuestra base de datos lista para ser usada por su interfaz web como por ejemplo con Angularjs o Ionic Framework. Por todo esto sólo necesitarán un cliente que se comunique con JSON para acceder a la capa ORM de abstracción de la base de datos.

En nuestro caso, la base de datos será nuestra carta más importante ya que todos nuestros sensores son quienes alimentan y llenan la base de datos así que de alguna forma es la actividad del usuario con esos sensores la que se puede aprovechar por aplicaciones de terceros.

Por lo tanto, una API REST, básicamente, lo que hace es crear una capa intermedia entre la base de datos y el cliente, haciendo de manera sencilla el hecho de acceder a los datos de la aplicación mediante métodos comunes en entornos web como pueden ser GET, POST, Update, etc...

Dejando también a un lado SOA, tenemos los Web Sockets como estrella para nuestras comunicaciones. Lo más estándar en el mercado es utilizar SIP como protocolo de sesión junto con RTP y RTCP en el transporte basado en UDP. Todo esto lo podemos hacer en Javascript o en Python con sockets.io o crossbar.io. En la etapa de desarrollo se implementará este sistema para que haya comunicación full-duplex o bidireccional con los usuarios o clientes que lo remitan. La gran diferencia entre éstos y otros métodos de comunicación es el modelo Push de datos del servidor al cliente, frente al modelo Pull del cliente al servidor.



4.6- Servicio Reconocimiento de Voz Remoto

Para plantear un sistema de reconocimiento de voz tenemos básicamente dos opciones. La primera es realizar el proceso en local con un servidor dedicado para ello. Pero la tendencia dentro de Internet es delegar este proceso a una plataforma especializada.

Así, hemos elegido una plataforma que ofrece servicios biométricos de voz y cara en la nube. Ésta plataforma es accesible mediante REST, lo que nos permite mantener nuestra propia arquitectura interna abstrayendo todo el proceso biométrico. Desde nuestro núcleo de ProceSiX.

Hemos valorado delegar el proceso tanto a la plataforma que ofrece Google como a la que ofrece Facebook, pero nos hemos decidido por una plataforma específica llamada BioID, que como hemos dicho ofrece biometría en la nube. Con esto tendremos un sistema biométrico accesible desde Internet mediante REST, por lo que podremos interactuar con la base de datos biométrica.

Este servicio es muy importante, y obviamente si hubiéramos trabajado con Android habríamos utilizado el sencillo sistema de voz de Google o si fuera el caso de Apple sería Siri o

alguna aplicación similar. Nuestra elección es de algún modo con una plataforma neutral que hace más seguro los datos de la comunicación.

4.7- Seguridad Web y ERP

Debido a la sensibilidad de los datos que ofrece la plataforma, como los biométricos, hemos elegido hacer una auditoría de seguridad sobre la red local que forma el prototipo.

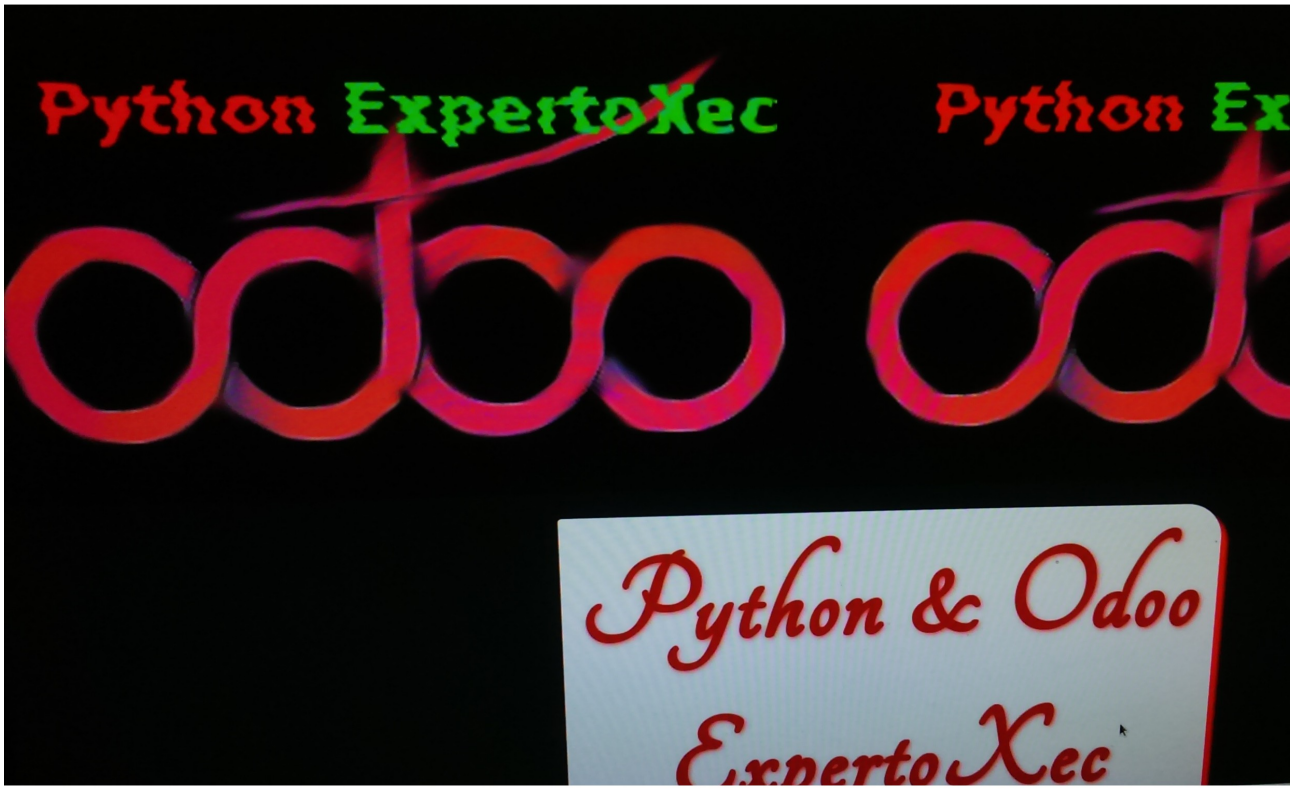
Es una prueba concepto básica centrada sobretodo en los servicios web pero sin olvidar por ejemplo SSH. Para comparar los resultados de nuestros servicios con alguna referencia vamos a realizar la misma auditoría a un servidor con Odoo (OpenERP) y Django remoto contrastándolo con nuestro prototipo en red local.

La prueba va a consistir en emular un intento de penetración en el sistema por parte de un computador en la red local para en adelante poder hacer esta actividad continuamente y así estar en busca de vulnerabilidades en tiempo real, llamado pentesting continuo. En esta prueba se van a seguir las fases establecidas en la teoría sobre seguridad ofensiva, un test de penetración. Principalmente nos vamos en centrar en analizar el sistema web ya que es lo más común hoy en día y el puerto 80 el único abierto en un servidor.

Para contrastar los resultados del análisis a nuestra plataforma vamos a analizar otra plataforma con un sistema Python formado por Gunicorn y Django y sus aplicaciones en el puerto 80 así como un ERP (Odoo v8 y v9) que tiene todo su funcionamiento en el puerto 8069 y es igualmente un sistema web que usa las mismas técnicas que cualquier otro.

Los resultados son que el ERP está bien asegurado mientras que nuestros servidores, cluster… tienen muchas vulnerabilidades conocidas. Ésta ofrece un contraste medible en términos de riesgos.

Con esto comprobamos que nuestro prototipo aún no está preparado para salir a Internet y sólo es seguro si es ejecutado por completo por nuestros propios equipos en nuestra propia red. Una vez en este entorno se haya conseguido un nivel alto de seguridad y bajo riesgo de ataque podrá ser usado en producción y en el futuro escalar hacia más usuarios.


Python ExpertoXec
Python Ex
Python & Odoo
ExpertoXec

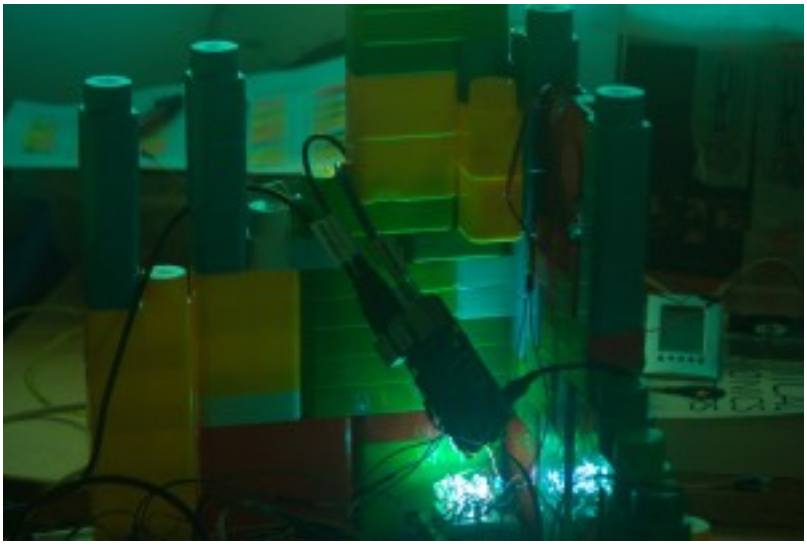
4.8- Computación Intensiva

Una prueba de concepto que no podía faltar es la de aprovechar la infraestructura de nuestro prototipo para realizar computación paralela o distribuida.

Debido a que todas las líneas apuntan a que se resuelvan problemas más complejos la necesidad de poner varios computadores actuando en conjunto para acelerar la respuesta y precisión del sistema es obvia.

En nuestro caso hemos configurado una Raspberry Pi como máster de un cluster o grupo y otra como nodo de computación esclavo. Una vez tenemos esto, incluir un mayor número de nodos de computación sólo es cuestión de las necesidades de la aplicación. Nuestro prototipo tiene 2 conmutadores y enrutadores con 4 puertos Ethernet, pero si quisiéramos expandir el poder de computación sólo tendríamos que añadir un conmutador con un mayor número de puertos que funcionaría de bus o troncal del cluster o bien hacerlo por WiFi.

En el correspondiente anexo detallamos los pasos necesarios para realizar la configuración mínima necesaria con la que obtendremos un cluster de dos Raspberry Pi calculando ejemplos como los decimales del número Pi. Quedando así demostrado que mediante MPI las máquinas se hablan y conforman una arquitectura distribuida y/o paralela, adecuado para el cálculo intensivo de datos como son los de nuestra aplicación accesibles mediante REST API y WebSockets.



Raspberry Pi 2 Master

4.9- Audio API y WebRTC

El primer requisito del audio es la baja latencia así como el bajo jitter o variabilidad del retardo. Con esto asegurado, tendremos lo que nos parecerá audio en tiempo real, con él podremos desarrollar videojuegos, salas de reuniones multimedia o como nuestro caso una estación completa de procesado de audio embebida en cualquier navegador moderno que hará las veces de consola digital o mesa de mezclas. Toda esta lógica está escrita en Javascript mediante el Web Audio API, pero podemos generar una cadena de compilación que traduzca código C/C++, OpenCL (C99), Python, Faust, PureData, Processing, Arduino al código estándar del estándar ECMScript que es representado por JavaScript en sus últimas versiones.

Con el Web Audio API podemos hacer cualquier procesado que se pueda también hacer en una mesa de mezclas profesional o equipo multipista. Por ejemplo, el WAAX (Web Audio API eXtension) es un software de última generación que abstrae muchos de los problemas que genera javascript, se podría decir que es como un framework para trabajar con Audio en la web directamente.

Además del Web Audio API también tenemos varias opciones con APIs de vídeo para su procesado. La idea es igual, obtener embebido en la web una mesa de mezclas digital para procesar imágenes o vídeo o televisión. La idea más cercana es utilizar la APIs webRTC para transmitir por Internet el vídeo procesado así como el Audio procesado. Además por webRTC también se pueden utilizar canales de datos que funcionan sobre TCP.

Para el webRTC tenemos las opciones de hacerlos con señalización SIP, el estándar de la telecomunicación, con XMPP de Google, P2P o Multicast. Nosotros hemos usado las que más se nos han adaptado a nuestras necesidades de las pruebas de concepto puestas en marcha. Realmente, las cuatro opciones han sido probadas para comparar la funcionalidad de cada solución así como la eficiencia de cada implementación.

Con todo esto, ahora nuestro objetivo es generar un Audio Streaming de bajo retardo para un sistema de audio 3D. El objetivo será grabar en multipista todos los flujos generados por nuestros instrumentos virtuales del prototipo como la guitarra y el cajón.

El audio 3D es transmitido por HDMI o por Jack Stereo así como por Ethernet o WiFi al otro extremo después de haber sido procesado apropiadamente.

El entorno linux es un Debian Jessie para Raspberry Pi 2 y consta de una capa ALSA, un OPUS encoder, una capa RTP y en trasporte UDP. Como resultado obtendremos un fichero a emitir con VLC en formato MPEG-4, como la Televisión Digital o en su defecto MPEG-2 como la TDT española. En el artículo “Low Delay Audio Stremaing for a 3D Audio Recording System” tenemos este ejemplo desarrollado con pruebas de concepto como el consumo de recursos de la CPU para los procesos de codificación y decodificación, los más importantes.

Por último, dejamos una fotografía de la antena WiFi helicoidal construida a mano tiempo atrás. Para que cualquier USB tenga WiFi y pueda enviar flujos multimedia a distacia por WebRTC tenemos esta antena con conversor a USB.

5-Conclusiones y Líneas Futuras

5.1- Conclusiones

Hemos obtenido un sistema capaz de percibir variables del entorno próximo y de exponer esta información al usuario de forma remota en la web o presencial mediante sus cinco sentidos.

También se ha integrado este sistema con una aplicación vertical que nos permite controlar el sistema de forma transparente y sencilla simplificando el sistema a “una cosa”, Thinger.

De esta manera hemos cumplido los objetivos marcados al principio del trabajo de establecer las bases de un sistema que nos permita integrar más elementos sobre él.

Elementos como una guitarra electroacústica con el sistema de sonido a medida implementado por Arduinos y Raspberry Pi, así como por una pantalla, teclado, ratón, disco duro, WiFi (Antena Helicoidal Maker), y los 6 micrófonos posicionados estratégimante en cuánto a acústica se refiere.

En cuanto a actuadores tenemos 4 altavoces construidos a mano como marca la filosofía maker o DIY.

Además, también se ha integrado un sistema de monitorización ambiental de humedad y temperatura analógica con dos sensores.

Todo esto está conectado con la red local y con Internet para cumplir con las necesidades actuales de tener conectado todas las cosas a una plataforma de gestión y monitorización como es Thinger, según lo comentado en el objetivo número 2.

Con todo lo conseguido al momento, tenemos una arquitectura de computación distribuida heterogénea; formada por: Raspberrys Pi, Arduinos, Ubuntus y DD-WRT.

Dentro de esta arquitectura tenemos varios stacks o pilas de aplicaciones funcionando como servidores. Estos stack son ya conocidos como LAMP, MEAN, Python, Docker, VMDK, JuJu, Ubuntu Studio, Processing, Arduino, IPFire, Elastix y Asterisk, OSSIM, Kali Linux, Metasploit, Ubuntu Server, Rune Audio OS, Raspbian Jessie, ZAP Proxy, Spark y Hadoop, mpi4py, Fortran, Faust, WAAX, SIPjs, EasywebRTC y Web Audio API. Concluyendo, con todos estos stacks hemos formado un núcleo o core heterogéneo de una aplicación llamada ProceSiX que interactúa con otra también vertical como es Thinger.

Thinger es para monitorización, comando y control. ProceSiX es para Análisis de Datos Masivos en Paralelo y en Distribuido, Inteligencia Artificial Aplicada y Robótica, Fusión de Sensores Multimedia, Inteligencia de Contexto, Domótica, Robótica Industrial, Bio-Tecnología, Inteligencia Ambiental, Sistemas Multi-Agente, Operaciones, Planificación, Navegación, Sistemas Embebidos y Procesado, Análisis, Simulación Acústica y Síntesis de Audio, Música y Voz.

Con todo este elenco de aplicaciones telemáticas y físicas más las partes de terceros, surge la necesidad de ofrecer formación sobre el uso y control de todos estos sistemas. Si tenemos una lavadora dando vueltas tenemos que saber usarla, así que escuchamos música relajante bio-inspirada también esta bien saber cómo funciona en realidad ese objeto y cómo nos transforma la realidad a una un poco mas virtual. Por ello es totalmente necesario el cumplir el objetivo de que este proyecto concluya con la capacidad de enseñar al resto de la sociedad lo aprendido en él.

Se han aprendido muchas tecnologías desde “Hello World” hasta algoritmos complejos embebidos en la web o en cluster heterogéneo. Se ha trabajado con una cadena de software enorme abordando diferentes lenguajes y sistemas informáticos aunque principalmente se ha hecho hincapié en Linux y en Python así como en REST, JSON y Web Sockets e Internet de las Cosas, Inteligencia Artificial y Robótica, Domótica, Música, Audio y Voz, Aplicaciones Híbridas Web y Móviles, Servidores completos incluyendo Librerías e Integración con otras Aplicaciones en los mismos Frameworks como son: Django, Flask, Bottle, Tornado, Gevent, Node.js, Angular.js, Express.js, PostgreSQL, MySQL, Cassandra, MongoDB, Crossbar.io, SIP.js, EasywebRTC, Elastix MT , Asterisk, OSSIM, Kali Linux, Ubuntu Server y Ubuntu Studio, Software de Edición Multimedia como: VLC, Gstreamer, VP8, Opus, MPEG-2 y MPEG-4, RTP, RTCP, UDP, ; ALL-IP, IP Multimedia Subsystem, Apache2, PHP5, Wordpress, Moodle, Odoo, OpenERP, MPI, Fortran, C/C++, OpenCL, MPI4PY, Cluster HPC Heterogéneo, Piezas de Juegos Infantiles, Juveniles y Adultos LEGO y MOLTO; para la enseñanza.

La educación en TIC es esencial hoy en día y representa el futuro trabajo de las generaciones que hoy están en el sistema educativo global. Puesto que el último objetivo es adquirir la capacidad de enseñar y seguir aprendiendo sobre todas las Aplicaciones, Frameworks y Lenguajes utilizados para construir sistemas complejos que algún día serán desde juguetes autónomos hasta cuidadores de personas con pocos recursos y capacidades.

En la red de dominios construidos en Internet existe ya un curso de Aplicaciones Ubicuas puesto en producción que engloba varias especializaciones y una base troncal de adquisición de conocimientos y competencias profesionales para los más jóvenes, a partir de 8 años en adelante pueden entrar en “www.multitic.es/escuela” y contactar para acceder al curso virtual preparado con toda la documentación expuesta en la amplia bibliografía que tenemos más adelante.

Para adentrarse en este mundo hace falta tener corazón de “maker” , “geek”, ”nerd”, ”alma curiosa”, ”corazón luchador” ,”flexible”, “resiliente”, y con muchas ganas de aprender y avanzar trabajando muy duro por un futuro a través de un presente mejor cada día del mes del año.

5.2- Líneas Futuras

Para nosotros, este apartado es el más importante, pues la idea del trabajo es dar cabida a soluciones más complejas de la forma más inmediata posible. A pesar de ello, seguro que se nos olvidan muchas aplicaciones posibles que podrían salir de la raíz de este trabajo debido a las inmensas posibilidades de desarrollos que existen. Por ello, vamos a ver bastantes posibilidades futuras a las que da lugar este trabajo:

- La primera opción a valorar en desarrollos futuros, para nosotros, sería integrar un mayor número de sensores y actuadores en el sistema, dando lugar a una capa lógica capaz de fusionar la información obtenida de todos los sensores y actuando en consecuencia dependiendo del uso. Con ello podríamos empezar a hablar de una plataforma de inteligencia ambiental aplicable tanto a domótica como a aplicaciones a campo abierto ya sean para monitorización de la naturaleza para prevención de riesgos naturales o de efectos humanos, o para aplicaciones de supervisión en entornos urbanos, en entornos de guerra u hostiles, etc

- La segunda opción que más nos llama la atención es la de aumentar el número de computadores en cliente, formando pequeños grupos para computación paralela o distribuida y consiguiendo mayor poder de computación in situ y dejando los centros de control remoto únicamente para la supervisión del sistema y librándolos de carga de trabajo. Esto es necesario pues para aplicaciones críticas que necesiten tomar decisiones y actuar en consecuencia sin intermediar con el centro de control, se necesitan ejecutar algoritmos complejos que no permiten que haya retardo o variabilidad del mismo introducido por la red. Podremos instalar hasta 64 nodos Raspberry Pi 2 e interminables Arduinos así como ATXs.

- El impacto que pueden llegar a tener estas aplicaciones que toman datos y actúan en entornos típicamente humanos puede llegar a ser devastador. Por ello, otra línea futura es evaluar los términos asociados a seguridad de la información, riesgos TI, así como la fiabilidad y precisión del sistema. Este paso sería esencial para pasar a producción las aplicaciones que surjan de este sistema o prototipo.

- El Armario Empotrado con ruedas es como un rack o armario metálico profesional pero de madera de pino. En el presente, dentro del mueble con ruedas hay muchos dispositivos o partes como son los informáticos y electrónicos así como electroacústicos y acústicos. Además tenemos todo el cableado eléctrico y electrónico. Como línea futura ya tenemos un plano de un nuevo armario en vez de mueble, con unas dimensiones de 1,40 x 1 x 0,60 metros que puede servir como un electrodoméstico más que como hemos visto depende de las piezas que lo integren así saldrá el presupuesto. Con 600 € se tiene un puesto informático didáctico y práctico para las tareas de la casa sobre todo las multimedia y robóticas pero que incluyen tanto la inteligencia artificial asociada como los conceptos del Internet de las Cosas en el mismo armario. Fundamental para la educación formal y no formal, en institutos o universidades y escuelas de las nuevas generaciones.

- Otra línea futura es construir una guitarra totalmente eléctrica con las arquitecturas y hardware visto hasta el momento, ayudados por un profesional de la madera. También se puede añadir el hardware de computación al cajón flamenco y al Djembe que tenemos como partes secundarias de la enseñanza musical al ser ambas percusivas y tener otras características acústicas. También probaremos el sonido con voz cantada y con una armónica, lo cuál todo junto forma una banda de música profesional que se puede alinear con las filosofías vistas como las DIY, DRY, pythonic way, javascript way, processing way y arduino ways.

- Como línea complementaria a la anterior, vamos a usar el software de simulación multi-física con la especialidad en mecánica y sobre todo acústica, COMSOL. Tiene licencia de pago pero con la versión de pruebas nos será suficiente para simular el comportamiento acústicos de los instrumentos virtuales, tendremos un CAD de dibujo junto a un CAD de electrónica, mecánica,

acústica, mecánica de fluidos y electroacústica todo en un mismo paquete software Java. Una vez tengamos la simulación, el proceso de impresión 3D es directo pudiendo construir instrumentos de madera, metal, vidrio, plásticos, y el resto de nuevos materiales para las tecnologías de la información y las comunicaciones.

-Recientemente, tenemos un osciloscoio en el taller. Con éste osciloscopio podremos analizar y posteriormente y simular con COMSOL o Simulink, todos nuestros instrumentos, desde la antena helicoidal hasta los arrays de micrófonos y altavoces pasando por la guitarra, cajón y armónica.

- Con todas estas herramientas tenemos suficiente justificación para llegados a este punto utilizar la cámara anecóica de la universidad como colaboración en este proyecto en futuro. En ella tenemos un armario metálico profesional, una mesa de mezclas semi analógica, la propia cámara anecóica con sus micrófonos y altavoces de medida que son perfectos para analizar los comportamientos de nuestros instrumentos virtuales, recordemos: Voz, Armónica, Guitarra Española Electroacústica, Altavoces Estéreo, Cajón Flamenco y Djembe Africano.

- Otra línea, es cambiar el audio por el vídeo, al asignar recursos de computo así como al orientar las aplicaciones posibles aplicadas con inteligencia artificial. En este tema el GIAA, tiene serios avances así como en sistemas multi-agente, minería de datos, fusión de sensores, técnicas de estimación modernas, redes neuronales, algoritmos genéticos, técnicas de clasificación máquina y soporte a la decisión y tratamiento digital de señales avanzado, y al fin un “saber hacer” en el ámbito del vídeo aplicado a vigilancia, navegación , reconocimiento, monitorización y guiado autónomo, lo que sería una lista de líneas futuras para las que desarrollar con el mismo sistema de audio, optimizarlo para vídeo, o aplicar las técnicas de vídeo al audio.

- También con el propio departamento tenemos la línea futura de embarcar en un dron una Raspberry Pi y procesar el vídeo u otra información de otros sensores con nuestra plataforma ProceSiX enlazada con Thinger, haciendo del Dron un robot autónomo como cosa del internet de las cosas.

- En éste dron se puede embarcar unas cámaras de Infrarrojos, que darán como resultado aplicaciones para la monitorización, gestión, control, y operación en entornos medioambientales como montes o riveras y mares.

- Ahora, sobre investigación de arquitectura de computadores, otra futura línea sería implementar el software en su totalidad en OpenCL y C11 y algún híbrido más. Ésta línea representa la de mayor peso ya que se trata de traducir todos los lenguajes utilizados hasta el momento a uno o dos como son C99 o C11, para que se puedan aprovechar las capacidades paralelas en GPU y distribuidas en diferentes CPU con implementación de MPI y Ethernet y SSD masivo, optimización total. Lo ideal sería primero hacer un análisis de viabilidad simulando los costes de traducir el sistema en python , javascript, processing, java, c, c++ a una implementación pura en el nuevo estándar OpenCL (Open Computing Languaje).

-Las arquitecturas están esquematizadas pero no están simuladas físicamente ni virtualmente, luego tendríamos que aplicar un proceso de selección de desarrolladores que se encargaran de todo el proceso de implementación con metodologías ágiles con Scrum y Kanban

como referencia. Para ello ya tenemos diagramas similares a UML para crear los proyectos con orientación a objetos. La implementación implicaría mucha más carga horaria que las demás líneas porque además unificaría todas las líneas de implementación del proyecto y por tanto abarcaría a unas 10 personas 8 horas diarias con metodologías flexibles durante un largo periodo de tiempo.

- Las arquitecturas muestran en detalle en nivel de sistema de las aplicaciones compatibles. Tenemos tanto el sistema general como los subsistemas informáticos, electrónicos y eléctricos detallados. Con explicaciones teóricas podremos explicar y dar formación sobre todos los sistemas y subsistemas del sistema, El Equipo.

- Otra tarea futura es implementar en UML las arquitecturas funcionales y técnicas para poder desarrollar con orientación a objetos. De nuevo la tarea mas laboriosa que nos falta es un buen desarrollo con su correspondiente optimización.

- El kernel, núcleo o core es la esencia, el corazón software del proyecto. Las extremidades y sentidos son los sensores, actuadores y personas y construcciones artificiales, como sillas de ruedas. La cabeza del software es funcional y en ella se describe los procesos funcionales que tienen lugar allí, de manera secuencial o algorítmica. Aunque vayamos a utilizar orientación a objetos las instrucciones máquina una vez hemos compilado son secuenciales y por lo tanto computables de manera distribuida, masiva y paralela y en tiempo real o con requisitos de tiempo para el multimedia. El núcleo ha de estar escrito en C++, concretamente en C11 y también en C específicamente en OpenCL o C99.

- La shell o caparazón sería como nuestras armaduras, cubren todas las extremidades, cabeza, sensores y actuadores, y sobre todo interactúan con el corazón y los pulmones así como con el resto de órganos internos para protegerlos y mantenerlos a temperatura y química adecuada y ante todo conectados unos con otros por el sistema nervioso, respiratorio, cerebral, sanguíneo, celular, etc... Este caparazón va a ir implementado en Python, con las partes que interactúen con el núcleo escritas en Ctypes o Cython con una cadena especial de software enfocada a la eficiencia pero siempre dentro de Python que podemos ver en la parte de Profiling de la bibliografía. Simplemente podemos medir también los tiempos de interacción con el kernel así como de transmisión, conexión, comunicación, tuberías, redirecciones, procesos, memoria ram y rom así como las memorias caché y todos los cuellos de botella que se puedan formar por la escasez de algún recurso utilizado.

- Las librerías son como la biblioteca o el conocimiento de hacer y saber cosas útiles, teóricas o experimentales. Es parte fundamental de cualquier red, en este caso red de ordenadores, cosas y objetos de código máquina. Podemos optimizar las librerías hasta el nivel del lenguaje ensamblador pero esto sólo será útil en los procesos de codificación y decodificación multimedia así como en ciertas partes del núcleo.

- Como siguiente línea futura, tenemos estudiar mediante sensores adecuados la respuesta humana a una señal compleja de música, pudiendo llegar a saber como es la cognición, percepción y el lenguaje verbal humano, ya que éste es muy similar estructuralmente al lenguaje musical. Además ambos tienen espectro de potencia distribuido de algún modo según los parámetros y la calibración de los dispositivos de medida y análisis por el que podemos conocer todos los detalles de la señal a nivel físico y matemático o estadístico. En cualquier caso con herramientas como la lógica de predicados formal traducida con, la teoría del información, la

cibernética, inteligencia artificial, de la probabilidad y con lógica formal filosófica y psicológica además de antropológica al ser la música un lenguaje transcultural.

-Con el conocimiento que hemos adquirido sobre algoritmos bio-inspirados aplicados a la solución de problemas de la realidad podemos hacer cantidad de ejercicios como pruebas de concepto sobre música pero es bueno saber que junto nuestra anterior línea sobre procesado de vídeo multi-cámara podemos, con los mismos algoritmos aplicados a la música aplicarlos a detección de movimiento en vídeo así como análisis de imágenes de alta resolución, multi-espectrales, de infrarrojos, HDR, etc... Y por supuesto, para entender las técnicas bio-inspiradas es necesario conocer las técnicas clásicas por lo que éstas también serán aplicables directamente al vídeo, como el filtro Wiener o los fractales para codificación multi-capa.

- También será un futuro desarrollo el desarrollar un tutorial web con WAAX sobre Procesado de Sistemas Audio, en el que desarrollar ejemplos que puedan funcionar en la sandbox de los navegadores mediante una cadena de software que resulte en JavaScript.

- Nuestra aplicación ProceSiX, brinda las capacidades de computo para este tipo de aplicaciones, fusión de sensores, así como un bajo coste de producción y de hardware aunque una alta sofisticación software con lenguajes y sistemas informáticos de alta dimensiones.

- En vez de radares, otra línea es hacer la fusión de datos e información con micrófonos como los nuestros. En el prototipo ya tenemos el chasis para los arrays de micrófonos. Están empotrados con una base de madera de okume a la misma guitarra, cajón o Djembe. Para calibrar el sistema únicamente hay que posicionar un tornillo pequeño para empotrar el micrófono u cualquier otro componente y mover los potenciómetros correspondientes para manipular los niveles de la señal y de los umbrales o triggers.

- Otra aplicación interesante que surge a raíz del proyecto, es utilizar el sensor de ultrasonido para entornos submarinos para detección de torpedos, materiales artificiales, submarinos o seres vivos del hábitat submarino.

-Las simulaciones en todas estas aplicaciones son extremadamente importantes de hacer para evaluar los costes y la viabilidad del proyecto así como para hacer el plan específico con todos los detalles medidos y computados por ordenador en digital. Después en el medio analógico tendremos que hacer pruebas y medir la respuesta específica del entorno y así poder comparar con el modelo digital.

- Los sensores infrarrojos se utilizan mucho en aviónica, los modelos de turbulencias caóticos también, la aeronáutica tiene múltiples aplicaciones desplegables desde nuestra plataforma ProceSiX. Sensores de todo tipo, simulaciones software multi físicas y laboratorio de matrices avanzado. Los aviones comerciales son sistemas expertos que han aprendido porque les han enseñado a pilotar el avión hasta incluso en las condiciones más adversas e impredecibles. Desde el satélite y GPS de navegación y planificación hasta el control de las turbinas, de la presión atmosférica, seguridad ante desastres, prevención de riesgos laborales, y mantenimiento de la salud pública del recinto aislado como son las diferentes sub-cámaras que componen las aeronaves.

- Los aviones mencionados utilizan satélites como por ejemplo el ultimo prototipo

con Raspberry Pi, en cluster, con paneles solares, o también la versión de Arduino, ArduSat.

- Otra aplicación estratégica es la biomedicina, la salud pública, desde los rayos X hasta la lucha contra el cáncer con métodos analíticos o la experimentación psico-social mediante las redes sociales o antropológica y filosófica con todas sus herramientas. En especial daremos prioridad a una aplicación de musicoterapia en directo o grabada y asistida por un ProceSiX comandado por un Doctor en Medicina experto en salud con musicoterapia. También hay mucho más ámbitos de desarrollo como predecir los infartos mediante arritmias, controlar las células, la tensión, la temperatura, la localización GPS, todos son de emergencia o de monitorización.

- La realidad virtual y aumentada tienen aplicaciones desde los juegos inmersivos hasta las gafas de Google, la Kinnect, la Wii, la PSP, pero también tienen aplicaciones médicas como estudiar neurociencias y psicoanálisis mediante el comportamiento observado en los sujetos mientras interaccionan con sus asistentes o con su otra realidad ya sea virtual o aumentada. Esto da lugar a un conexión dedicada que integra la Neurociencia con el Psicoanálisis y trata de potenciar la intersección entre ambas disciplinas de la ciencia.

- Todas estas aplicaciones serán soportadas por varios servidores distribuidos geográficamente, además de los prototipos que conformarán un BackEnd para Big Data ya que las aplicaciones que estamos contemplando utilizan una inmensidad de datos y además deben consumir pocos recursos en términos de costes globales.

- Las comunicaciones unificadas están basadas en la Voz IP pero tienen más canales y medios para realizar la comunicación. En esta línea se trata de poner en producción un sistema de Streaming de vídeo y audio con multicast, P2P, WebRTC, Linux low latency kernel y Asterisk o Elastix para WebSockets con señalización posiblemente SIP integrada en IMS. Este sistema es virtual y se encuentra el el MASTER.

- En cuanto a seguridad, hemos hecho una línea muy potente para tener un diseño basado en la seguridad, por ello las herramientas que utilicemos han de estar al altura. En el futuro el pentesting ha de ser un demonio instalado y no un mero cliente ejecutando programas especiales. Entre estos programas destacan Latch, Metasploit, Kali Linux, Wireshark, BeEF, W3AF, Nikto, Assaultjs, VAST, OSSIM, Nagios, WAF y filtros, Kali Linux e IPFire y pfSense.

- En cuanto al Internet de las Cosas, integraremos en su totalidad el sistema con Thinger y además ofrecemos a su desarrollador la posibilidad de documentar las partes que usemos en las líneas futuras. Las primeras líneas son enlazar el sistema con Android y acceder al kernel de Linux desde la plataforma remota distribuida en Amazon.

- Lo único físico que sólo es mecánico y no tiene nada de electrónica es el péndulo de 3 grados de libertad. Representa un sistema no lineal o con comportamiento caótico o sea impredecible a corto ni a largo plazo. Sólo cuándo éste esté en su régimen de funcionamiento podrá ser predecible, sino nos enfrentaremos a fenómenos como los fractales. En el prototipo tenemos una construcción con el Mecano con la estructura pendular, pero para que los ejes tengan buena rotación se necesitaría una construcción mucho más elaborada. Para esta línea estamos inspirados en el péndulo de 2 grados de libertad que muestra en sus intervenciones el “padre” de la teoría del Caos.

- Un Generador Pseudo-Aleatorio por hardware con Raspberry Pi y Python por software para obtener claves para SSH personales y certificados digitales firmados digitalmente. Sería el módulo del core para criptografía con el que obtener semillas pseudo aleatorias para cifrar todos nuestra capa de información verticalmente así como las comunicaciones y túneles como VPN.

- Y la siguiente aplicación que es algo físico, es la agro-tecnología, con servicios telemáticos que representan el Internet de las cosas con maquinaria de la industria agrícola, de montes y de vida oceánica. Todo el medio-ambiente será monitorizado con cosas para prevenir catástrofes, controlar los niveles de componentes químicos, de contaminación, control del tiempo atmosférico, control de los caudales de los ríos, de los diques de las costas, de las playas y acabamos de las ciudades dónde el ahorro energético será clave en todos los edificios de las Smart Cities, dónde acaba el campo.

-Hasta ahora tenemos todas las principales vías de desarrollo del sistema que se resumen en aumentar la capacidad de adquisición de datos y actuación, la capacidad de computo en los extremos y la seguridad de los procesos que se ejecutan en el sistema así como todas las múltiples aplicaciones y proyectos completos que aún quedan por desarrollar y estamos presentando. Además, en los anexos correspondientes hemos planteado la forma de implementar estas soluciones para hacerlas un poco más presentes en el proyecto en vez de futuras, aunque ciertamente aún quedan muchos detalles a solucionar para poner en práctica éstas soluciones.

- Como ya hemos comentado anteriormente, las aplicaciones pueden llegar a ser infinitas pero desde luego hay que tener en cuenta cuales son más viables a corto plazo y cuáles forman parte de una realidad más cercana de la ficción, por esto tan importante la etapa de simulación y análisis previa.

- Para tocar la línea de la ficción simplemente vamos a imaginar que estamos tratando con un sistema robótico autónomo y móvil capaz de emular los sentidos humanos y de actuar en función de esta percepción. En términos más exactos, sería lo que se define como robot humanoide. Y una vez tenemos un robot humanoide, éste se debería integrar en nuestra sociedad, siendo capaz de interactuar con otros humanos o con otros robots, como los que recorren Internet. En el lado de servidor dispondríamos de un HPC de computación que funcionaria como asistente virtual tanto para humanos como para el propio robot humanoide.

-Ya que los robots humanoides, y sobretodo los asistentes virtuales están al alcance de pocos capitales u organizaciones, personalmente pienso que el objetivo a conseguir en el futuro más importante es el de democratizar poniendo al alcance del mayor número de personas posibles este tipo de sistemas. Las bases que se han utilizado son accesibles a un gran numero de organizaciones ya que se han utilizado sensores, computadores y actuadores de bajo coste, pudiendo llegar a acelerar el tiempo de acceso al mercado.

-Y bien, como estaba diciendo, el objetivo más importante es el de integrar en la educación en fases tempranas este tipo de sistemas potenciando así la imaginación y pragmatismo de cada persona. Y ya que queremos robot autónomos y asistentes virtuales pienso que en última instancia el mejor de ellos somos nosotros mismos, así que potenciar el “Hazlo Tú Mismo” sería el mayor paso para la revolución post industrial basada en el conocimiento. Nosotros pondremos todo nuestro material, ganas e ilusión en construir nuevos proyectos Hardware y Software libre para

niños , jóvenes y adultos. Nuestra manera de compartir el conocimiento adquirido en este proyecto será mediante páginas web, aplicaciones web y blogs o redes sociales con diferentes nombres comerciales y nombres de dominio.

-Como último objetivo estaríamos encantados de poder formar un grupo o comunidad de trabajo que permita continuar con todas las líneas futuras presentadas aquí ofreciendo formación gratuita y soporte en la nube para empresas.

6- Gestión del Proyecto

Recurso - Actividad	Tiempo (Horas)* *	Dinero (Euros)*
Planteamiento Arquitectura	200	4.000
Investigación Teoría	400	8.000
Desarrollo Software	100	2.000
Bibliografía	300	2.500
Materiales Electrónicos	100	500
Materiales Informáticos	100	3.000
TOTAL	**1.200 Horas**	**20.000 Euros**

*Es el trabajo total de 2 años.
*El tiempo y el dinero es una aproximación basada en una estimación personal.
*Cada hora corresponde a 20 € en las tres primeras actividades.
*En las siguientes tres filas, los recursos bibliográficos y materiales, las horas corresponden al tiempo gastado en la adquisición del conocimiento necesario para incluir los recursos en el proyecto. El dinero es una aproximación de lo que cuesta cada recurso en sí físcamente.

7-Bibliografía

7.1- Documentación:

7.1.1- Internet:

1- www.google.com

2- www.wikipedia.org

3- www.thinger.io

4- www.Arduino.cc

7.1.2- Electrónica, Acústica y Electroacústica:

1- ***Electroacústica Práctica.*** *Jan Voetmann y Eddy Bogh Brixen. Tebar. 2013.*

2- ***Practical Electronics for Inventors.*** *Paul Scherz y Simon Monk. McGraw Hill. 2013.*

3- ***Ingeniería Acústica.*** *Manuel Recuero López. Universidad Politécnica de Madrid. 2000.*

4- ***The Science & Applications of Acoustics.*** *Daniel R. Reachel. Spinger. 2006.*

5- ***Transductores, Micrófonos y Altavoces.*** *Manuel Recuero López. Universidad Politécnica de Madrid. 2000.*

6- ***Sistemas Audiovisuales. Televisión Analógica y Digital.*** *Francesc Tarrés Ruiz. Edicions UPC. 2000.*

7- ***Apuntes: "Tratamiento Digital de la Voz".*** *Departamento de Teoría de la Señal y Comunicaciones. Grupo de Procesado Multimedia. 2007-2008.*

8- ***Modelling the Sound Radiation by Loudspeaker Cabinets.*** *Cobianchi & Rousseau. B&W Group Ltd, England*

9- ***A Study into the Acoustic and Vibrational Effects of Carbon Fiber Reinforced Plastic as a Sole Manufacturing Material for Acoustic Guitars.*** *O'Donell, McRobbie. University of the West of*

Scotland.

7.1.3- Música, Movimiento, Lenguaje Natural, Psicología e Inteligencia Artificial BioInspirada:

*1-**Psicología e Inteligencia Artificial.** José Luis Zaccagnini. Pablo Adarraga. Trotta. 1994*

*2-**Composición Musical Bio-Inspirada.** Alberto Carretero. Punto Rojo. 2014*

*3- **The Nature of Code. Simulating Natural Systems with Processing.** Edited by Daniel Shiffman. 2012.*

*4- **Fundamentos de Inteligencia Artificial. Inteligencia Artificial Bio-Inspirada.** Antonio Benítez. Escolar y Mayo Editores SL. 2013.*

*5- **La Mente Musical: La psicología Cognitiva de la Música.** John A. Sloboda. Machado Grupo de Distribución SL. 2012.*

*6- **Music Perception and Cognition.** Andrés Pérez López. Music Technology Goup. Universitat Pompeu i Fabra. 2013*

*7- **Acústica y Psicoacústica de la Música.** Juan G. Roederer. Melos.2007.*

7.1.4- Inteligencia en Sistemas y Robótica:

*1- **Multisensor Data Fusion.** Edward Waltz y James Llinas. Artech Hause Inc. 1990.*

*2- **Ingeniería de Control. Modelado y Control de Sistemas Dinámicos.** Luis Moreno, Santiago Garrido y Carlos Balaguer. Ariel Ciencia. 2003.*

*3- **Robótica Móvil. Principios, Tendencias Y Aplicaciones.** Editado por Tomas de J. Mateo Sanguino. Universidad de Huelva. 2014.*

*4- **Robot Brains. Circuit & Systems**. Biblioteca UC3M Leganés. 2015*

*5- **ROS: Robot Operating System**. Jit Ray Chowdhury. AGV KGP.*

7.1.5- Audio & Music Programming Real Time:

*1- **The Audio Programming Book.** Edited by Richard Boulanger y Victor Lazzarini. 2011.*

*2- **Real-Time Digital Signal Processing.** Nasser Kehtarnavaz. Elsevier. 2004*

*3- **Apuntes: "Tratamiento Digital del Audio" y "Audio En Telecomunicaciones".** 2007-2008. Departamento de Teoría de la Señal y Comunicaciones. Grupo de Procesado Multimedia.*

*4- **Signal Processing for Music Analysis.** Müller, Ellis, Klapuri, Richard. IEEE. 2011.*

*5- **Real Sound Synthesis.** Biblioteca UC3M Leganés. 2015*

*6- **Python for Audio Signal Processing**. Glover, Lazzarini y Timoney. The Sound & Digital Music Research Group. University Ireland.*

*7- **Web Audio API W3C**. September 2015.*

*8- **Signal Processing Libraries for Faust.** Julius Smith. Center for Computer Research in Music & Acoustics (CCRMA). University Stanford.*

*9- **WAAX: Web Audio API eXtension.** Hongchan Choi, Center for Computer Research in Music & Acoustics (CCRMA). Jonathan Berger. CCRMA. Stanford University.*

*10- **Web Sockets Documentation Oficial.** Aymeric Augustin. Mayo-29-2015.*

*11- **Low Delay Audio Streaming for a 3D Audio Recording System.** Marzena Malczewska, Tomasz Zernicki & Piotr Szczechowiak. Zylia SP.zo.o.Poland.*

7.1.6- Programación Sistemas Altas Prestaciones y Embebidos:

*1- **Programación C/C++.** Alejandro Sierra Urrecho y Manuel Alfonseca Moreno.Multimedia Anaya. 2014.*

*2- **OpenCL Programming Guide.** Pat Hanrahan. Addison Wisley. 2011.*

*3- **High Performance Linux Cluster.** Joseph D. Sloan. O'Really. 2004.*

*4- **Building Embedded Linux Systems.** Yaghmour, Masters, Ben-Yossef y Gerum. O'Reilly. 2003.*

*5- **Desing of Hardware / Software Embedded Systems.** Eugenio Villar. Universidad de Cantabria. 2001.*

6- ***Programación Shell en Unix/ Linux.*** *Christine Deffaix Rémy. Eni. 2012.*

7- ***Practical Data Science Cookbook.*** *Tony Ojeda y Benjamin Bengfort y Patrick Murphy y Abhijit Dasgupta. Packt Publishing. 2014.*

8- ***ROS: Robot Operating System.*** *Jit Ray Chowdhury. AGV KGP.*

7.1.7- Profiling & Performance:

1- ***High Performance Python.*** *Micha Gorelick e Ian Ozsvald. O'Reilly. 2014*

7.1.8- Sistemas Operativos y Redes Informáticas:

1- Guía ***de Estudio CCNA Routing & Switching.*** *Ernesto Ariganello. Ra-Ma. 2013*

2- Guía ***de Estudio CCNA Security****. Ernesto Ariganello.Ra-Ma. 2014.*

3- ***Redes Cisco CCNP a Fondo.*** *Ernesto Ariganello y Enrique Barrientos Sevilla. Ra-Ma. 2010.*

5- ***Linux. Preparación para la certificación LPIC 1 y 2****. Sébastien Bobillier. Eni. 2013.*

4- ***Absolute OpenBSD.*** *Michael W. Lucas. No Starch press. 2013.*

5- ***CentOS 6 Linux Server Cookbook.*** *Jonathan Hobson. Packt Publishing. 2013.*

7.1.9- Internet de las Cosas con Raspberry Pi y Arduino:

1- ***Raspberry Pi. 200 Ejercicios Prácticos.*** *Simon Monk. O'Reilly. Anaya. 2015.*

2- ***Raspberry Pi Hacks.*** *Ruth Suehle y Tom Callaway. O'Reilly. 2014*

3- ***Sistemas Integrados con Arduino.*** *José Rafael Lajara Vizcaíno y José Pelegrí Sebastià. Marcombo Ediciones Técnicas. 2014.*

4- ***Una Panorámica de las Telecomunicaciones****. Anibal R. Figueiras. Prentice Hall. 2002.*

5- ***Learning Internet of Things****. Peter Waher. Packt Publishing. 2015.*

7.1.10- Web Frameworks, REST y WebRTC:

*0-**Two Scoops Of Django 1.8.** Greenfelds Brothers. 2015.*

*1- **RESTful Web APIs.**Richardson & Amundsen. O'Reilly. 2013*

*2- **Programming JavaScript Application**.Eric Elliot. O'Reilly. 2014.*

*3- **Learning Ionic. Build real-time and hybrid mobile applications withIonic.** Mike Hartington, Arvind Ravulavaru. Packt Publishing. 2015.*

*4- **ProDjango.** Marty Alchin. Apress. 2º Edition.*

*5- **Security Considerations for WebRTC draft-ietf-rtcweb-security-08.** 2015.*

*6- **Using ZRTP to Secure WebRTC draft-johnston-rtcweb-zrtp-02.** 2015.*

*7- **Transports for WebRTC draft-ietf-rtcweb-transports-09**. 2015.*

*8- **Web Real-Time Communication (WebRTC): Media Transport and Use of RTP draft-ietf-rtcweb-rtp-usage-25**. 2015.*

*9- **WebRTC Gateways draft-ietf-rtcweb-gateways-01**. 2015.*

*10- **WebRTC Video Processing and Codec Requirements** draft-ietf-rtcweb-video-06. 2015.*

*11- **WebRTC Audio Codec and Processing Requirements** draft-ietf-rtcweb-audio-08. 2015.*

*12- **Web RTC Audio APIs**. 2015.*

*13- **Introduction to javascript Object Notation**. Lindsay Bassett. O'Reilly. 2015.*

7.1.11-Seguridad Informática:

*1- **Metasploit para Pentesters.** Pablo González Pérez. Zeroxword Computing. 2014.*

*2- **Hardening de Servidores GNU/Linux.** Carlos Álvarez Martín y Pablo González Pérez. Zeroxword Computing. 2013.*

*3- **Pentesting con Kali.** Pablo González Pérez, Germán Sánchez Garcés y Jose Miguel Soriano de la Cámara. Zeroxword Computing. 2013.*

*4- **Ethical Hacking.** Pablo González Pérez. Zeroxword Computing. 2014.*

*5- **Linux Exploiting: Técnicas de explotación de vulnerabilidades en Linux para la creación de expliots.** David Puente Castro. Zeroxword Computing. 2013.*

*6- **Python para Pentesters.** Daniel Echeverri Montoya. Zeroxword Computing. 2014.*

*7- **Hacking con Python**. Daniel Echeverri Montoya. Zeroxword Computing. 2015.*

*8- **All In One Certified Ethical Hacker**. Matt Walker. Mcgraund Hill Education. 2014.*

*9- **Hacking y Seguridad VoIP**. Jose Luis Verdeguer Navarro. Zeroxword Computing. 2013.*

7.1.12- Odoo Open ERP:

*1- **Odoo Development Essentials**. Daniel Reis. Packt Publishing. 2015.*

7.1.13- Componentes del Sistema:

1- ADC+PGA: ADS1015 Adafruit 4 channel x1

2- Sensores Sonido x6

3- Sensor Ultrasonidos x1

4- Sensores Ambiental x2

5- Altavoces x4

6- Micrófonos x6

7- Micro Arduinox1

8- Wave Kit Audio Adafruit x1

9- Raspberry B+ x1

10- Raspberry Pi 2 x1

11- Asus eeePC 901 x1

12- Péndulo 3º Libertad Comportamiento Caótico x1

13- Relé Shield x4 x1

14- Caja MOLTO x1

15- Base LEGO x1

16- Interruptor x3

17- Casquillo x3

18- Bombilla x3

19- Ventilador x1

20- Cámara Web x2

21- Radio Despertador x1

22- Hub USB + Lector Tarjetas x1

23- NAS 2 TB x1

24- HDD USB 1TB x1

25- Micrófono Ordenador x1

26- Ordenador Personal ATX x1

27- Router Cisco WRT160LN x1

28- Router Linksys WRT 54 x1

29-Regleta Eléctrica x3

30- Guitarra, Cajón, Djembe y Armónica x1

8- Anexos

Anexo I: Raspberry Pi

1-Instalación Sistema Operativo en Raspberry Pi

-Para instalar el sistema operativo Raspbian disponemos de una memoria microSD de 8 GB con NOOBS preinstalado.

-Para instalar el sistema en Rpi 2 hemos de hacer lo siguiente (tener librería hal instalada) :

sudo apt-get install usb-imagewriter

-En nuestro caso hemos utilizado una microSD de 16 GB para instalar Raspbian. Descargamos la imagen oficial y la escribimos en la tarjeta con la aplicación instalada.

-Hemos de tener teclado y pantalla para realizar el proceso de instalación en ambos casos y es recomendable estar conectado a internet. Una vez instalado pasaremos a configurar el sistema operativo.

2- Configuración SO

-El primer paso a realizar es configurar la tarjeta de red con parámetros estáticos en /etc/network/interfaces editamos lo siguiente:

address [your chosen IP address]
netmask [your netmask]
network [your destination]
broadcast [your broadcast range]
gateway [your gateway]

-El segundo será el nombre de host en /etc/hostname y /etc/hosts.

-En cuanto al DNS:

/etc/resolv.conf:

nameserver xxx

-Lo tercero será entrar en la configuración y ponerla a medida:

sudo raspi-config

-Y cuarto, instalar algunas herramientas:

sudo apt-get install git wget

3- Servidor LAMP

-Para instalar la bien conocida pila LAMP:

sudo apt-get install apache2 php5 php5-mysql mysql-server

-Así, tendremos en /var/www/ los ficheros correspondientes a una página como la nuestra.

4- Servidor Django

-Lo primero será instalar las dependencias del entorno python:

sudo apt-get python-pip python-setuptools python-virtualenv python-dev python-software-properties

sudo pip install Django1.8

5- Servidor Bottle

-Para el servidor y librería Bottle:

sudo apt-get install python-bottle

6- Instalación Motion

-Para instalar el software que servirá el vídeo una vez detectado movimiento:

sudo apt-get install motion
En /etc/motion/motion.conf, editamos la línea a “daemon on” y la “webcam_localhost = off”
En /etc/default/motion, editamos poniendo “start_motion_daemon = yes”

-Con esto ya tenemos el servicio ejecutándose como demonio (una vez reiniciado) y por defecto servirá el vídeo en el puerto 8081.

8- Navegador Web

-Un navegador altamente recomendable para probar en la Raspberry nuestras pruebas de concepto es la versión de chrome para linux, en este caso armhf:

sudo apt-get install chromium-browser

9- GPIO

-Instalamos bibliotecas:

sudo apt-get install python-dev python-rpi.gpio python-smbus i2c-tools python-serial minicom

git clone git://github.com/doceme/py-spidev

- Y clonamos un repositorio que posteriormente nos será muy útil.

Sudo git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git

10- Instalación Thinger

- En la siguiente dirección tenemos las instrucciones para trabajar con Raspberry Pi junto a Thinger:

https://community.thinger.io/t/starting-with-the-raspberry-pi/36

11- Aplicación MEAN

Introduccion Aplicaciones híbridas, Angular, Nodejs, Express, MongoDB, Ionic, brunch

sudo apt-get install npm nodejs

git clone https://github.com/hyyan/brunch-with-ionic.git

-Install (if you don't have them):

- Node.js: brew install node on OS X
- Brunch: npm install -g brunch
- Bower: npm install -g bower
- Brunch plugins and Bower dependencies: npm install & bower install.

- Run:
- brunch watch --server watches the project with continuous rebuild. This will also launch HTTP server with pushState.

- brunch build --production builds minified project for production

- Learn:
  - www/ dir is fully auto-generated and served by HTTP server. Write your code in app/ dir.
  - static files in app/assets/ will be copied to www/.
  - Place styles in app/styles to be compiled to www/css/app.css
  - Place javascripts in app/scripts to be concatenated to www/js/app.js

--Anexo II: Arduino

-Con nuestra placa Arduino Micro, y el sensor de ultrasonidos, únicamente mediremos en el rango de los ultrasonidos la distancia, como ejemplo de otro sensor usado popularmente en robots móviles.

-En la siguiente dirección web tenemos el código necesario para hacer funcionar el ejemplo:

https://gist.github.com/flakas/3294829#file-hc-sr04-ino

-También con micro Arduino tenemos implementado el vúmetro ajustando mediante potenciómetro el umbral.

--Anexo III: Control de Luces

- Escudo de Relés forman una interfaz eléctrica en nuestro caso para tres bombillas pero podriamos enchufar directamente como un interruptor virtual cualquier electrodoméstico.

- Los diodos irán conectados directamente a los pines digitales y únicamente es necesario poner una resistencia en modo pull-up apropiada antes de conectar.

--Anexo IV: Control Acústico y de Audio

-El sistema de sonido está compuesto por un sensor de sonido o micrófono, por el ADS1015 que digitaliza las señales y por la propia Raspberry Pi. Como a lo largo de todo el proyecto no es nuestra necesidad implementar código desde cero ya que ya existen soluciones para nosotros.

https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code/blob/master/Adafruit_A

--Anexo V: Django_REST_RPC

1- Django

-Instalamos Django1.8 en el servidor una vez tenemos cumplidas las dependencias de Python:

sudo pip install Django1.8

- Creamos un proyecto:

sudo mkdir proyecto_django

cd proyecto_django

django-admin startproject proyecto1

- Configuramos la base de datos, lanzamos el servidor de pruebas y creamos una aplicación dentro del proyecto:

cd proyecto1

python manage.py migrate

python manage.py runserver 192.168.1.X:XX

python manage.py startapp aplicacion1

2- Framework REST

- Una vez tenemos el entorno Django preparado podemos añadir aplicaciones fácilmente. En este caso añadimos a la lista de aplicaciones el framework_rest, previa instalación del mismo e incluimos en las rutas la línea adecuada:

pip install djangorestframework

```
INSTALLED_APPS = (
    ...
    'rest_framework',
)

urlpatterns = [
    ...
    url(r'^api-auth/', include('rest_framework.urls', namespace='rest_framework'))
]
```

3- RPC 4 Django

-Añadimos la aplicación en settings.py igual que en el caso anterior y la ruta en urls.py:

```
INSTALLED_APPS = (
    'rpc4django',
)

urlpatterns = patterns('',
    # rpc4django will need to be in your Python path
    (r'^RPC2$', 'rpc4django.views.serve_rpc_request'),
)
```

--Anexo VI: Odoo v8

1- Instalación

-Para instalar el en el servidor linux remoto nuestro ERP como “cosa”, hacemos:

```
sudo git clone https://github.com/odoo/odoo.git -b 8.0
cd odoo
python odoo.py setup_deps
python odoo.py setup_pg
```

-Y lo ejecutamos con:

```
python odoo.py
```

2- Configuración

La configuración será mediante la interfaz web con menús intuitivos. Así, una vez hayamos creado

una base de datos podremos instalar todas las aplicaciones incluidas en el paquete básico que resultará en un ERP open source muy completo.

Tendremos odoo ejecutandose en el servidor sobre python2.7 expuesto en el puerto 8069 a Internet. Podremos autenticarnos en la web y utilizar el programa con privilegios de administrador.

3- Integración RPC

La integración es también a través del puerto 8069 y podemos utilizar XML o JSON con métodos RPC. Ésto lo haremos con rpc4django que detallaremos en el apartado correspondiente.

--Anexo VII: Computación Intensiva

1- Gestión de Colas Asíncrono y Entorno para Ciencia de Datos

sudo apt-get install python-numpy python-scipy python-matplotlib ipython ipython-notebook python-pandas python-sympy python-nose

sudo pip install Celery

2- Software de Control

sudo pip install Flower

3- Computación en Cluster con MPI

Con estas 2 herramientas tendremos la capacidad de formar un cluster de computadores con cualquier sistema operativo y arquitectura hardware que soporte Python. Por ejemplo, con Raspberrys.

Si queremos algo más serio nos podemos ir a Apache Spark que junto a Ubuntu resulta un sistema muy potente y actual, pero no va en la línea nuestra de Python aunque realmente hay adaptaciones para manejarlo con este lenguaje.

Pero falta detallar cómo sería la comunicación entre los nodos y el master, que será mediante el protocolo MPI. A continuación detallaremos el proceso completo para desplegar un cluster basado en MPI:

1º-Instalamos Fortran para poder ejecutar rutinas en éste lenguaje:

sudo apt-get install gfortran

2º- Compilamos MPI para nuestra arquitectura armhf de las Raspberrys:

mkdir /home/pi/mpich2
cd ~/mpich2

wget http://www.mcs.anl.gov/research/projects/mpich2/downloads/tarballs/1.4.1p1/mpich2-1.4.1p1.tar.gz

tar xfz mpich2-1.4.1p1.tar.gz

sudo mkdir /home/rpimpi/
sudo mkdir /home/rpimpi/mpich2-install

sudo mkdir /home/pi/mpich_build
cd /home/pi/mpich_build

sudo /home/pi/mpich2/mpich2-1.4.1p1/configure -prefix=/home/rpimpi/mpich2-install
sudo make
sudo make install

export PATH=$PATH:/home/rpimpi/mpich2-install/bin

-Añadimos al inicio del fichero .profile estas 2 líneas:

# Add MPI to path
PATH=”$PATH:/home/rpimpi/mpich2-install/bin”

-Para comprobar que todo funciona:

which mpicc
which mpiexec

mpiexec -f machinefile -n 1 hostname

- Y por último en el Master, ejecutamos el ejemplo que calcula el número Pi:

cd /home/pi/mpi_testing
mpiexec -f machinefile -n 2 ~/mpich_build/examples/cpi

-Hasta aquí hemos obtenido una imagen configurada con MPI que deberíamos clonar para tener una copia del master en Raspberry. Poniendo ésta imagen en una segunda Raspberry ya tendriamos 2 configuradas. Para simplificar el acceso entre ellas debemos ejecutar el la Master lo siguiente:

cd ~
ssh-keygen -t rsa -C “raspberrypi@raspberrypi”

-Habiendo ya puesto una frase de paso copiamos las claves a la segunda Raspberry:

cat ~/.ssh/id_rsa.pub | ssh pi@192.168.1.X “mkdir .ssh;cat >> .ssh/authorized_keys”

-Y, con esto tendríamos ya el cluster de 2 nodos pero si queremos usar Python antes de clonar de nuevo la imagen deberemos seguir los siguientes pasos:

sudo apt-get install python-mpi4py

-Para comprobar el funcionamiento podemos probar estos dos comandos:

mpirun.openmpi -np 2 -machinefile /home/pi/mpi_testing/machinefile python helloworld.py

mpiexec.openmpi -n 4 -machinefile /home/pi/mpi_testing/machinefile python helloworld.py

--Anexo VIII: Pentesting

1- Escaneo Pasivo y Activo

-Tenemos múltiples maneras de escanear los servidores, pero principalmente usaremos Nmap y Scapy para la red. En este caso, sólo damos el comando de instalación pues después el escaneo es obvio leyendo la documentación man y sobretodo es un proceso un tanto relativo ya que podemos ejecutarlo normalmente o en modo de evasión de medidas defensivas de seguridad.

sudo apt-get install nmap python-nmap zenmap python-scapy

2- Vulnerabilidades Web

-Al igual que en el escaneo dejamos instaladas las herramientas correspondientes que en su modo normal son fáciles de ejecutar:

sudo apt-get install nikto beef w3af
sudo apt-get install metasploit python-msfrpc
sudo apt-get install python-beautifulsoup python-mechanize

3- Explotación y Persistencia

En este paso, usaremos Metasploit para en el caso de explotar el sistema poder directamente lanzar

consolas directas o inversas para establecer la persistencia en el sistema objetivo. Puesto que anteriormente hemos instalado el plugin para acceder a metasploit desde python podremos interactuar mediante este lenguaje con todos los métodos del potente y conocido programa Metasploit.

4- Fortificación del Sistema

-Dependiendo del sistema con el que tratemos, lo primero que deberíamos hacer es cifrar el sistema de ficheros de la partición así como los datos, para después ya empezar a configurar el sistema para nuestras necesidades.

- Dos programas básicos para tener el sistema un poco atendido en cuanto a defensa continua son:

sudo apt-get install clamav rkhunter

-Configuramos rkhunter en /etc/rkhunter.conf y lo automatizamos con cron:

crontab -e

03 03 * * * /usr/bin/rkhunter --cronjob --update --quiet

-Suponiendo que disponemos de más servidores debemos tener también en cuenta otros programas tan útiles como conocidos pero que son un poquito más complejos en su configuración y dependen de nuestras necesidades.
-Así, por un lado tenemos Snort como IDS, en el que hay que tener mucho cuidado con las reglas que ponemos para que no interfiera con la funcionalidad de las aplicaciones que defiende.
-Y, otro software muy útil e incluso de moda a día de hoy es un SIEM, dónde la mejor elección basada en software abierto es OSSIM, que necesariamente debe de funcionar en un servidor a parte o quizás en uno basado en máquinas virtuales.

5- Fortificación del Servicio

-Primero, podemos utilizar el cliente en Python pareado con el software Latch. Sencillo pero muy importante tener un segundo factor de autenticación.

-Además, podemos enjaular todos los procesos o servicios para minimizar el espacio expuesto una vez un supuesto proceso fuese secuestrado.

- Por último, también podemos resaltar la creación de un RAT (Remote Administration Tool) para ser nosotros mismos los que tengamos el acceso garantizado a nuestros servicios.

-Ya que quizás este anexo parezca un poco complicado sobretodo al no tener instrucciones específicas recordar que en la bibliografía tenemos los documentos necesarios para implementar a medida nuestras políticas de seguridad, especialmente si las programamos con Python ya que los dos principales libros están basados en éste lenguaje.